IMS24

IMS and RRS working together to coordinate your updates

pete.sadler@uk.ibm.com

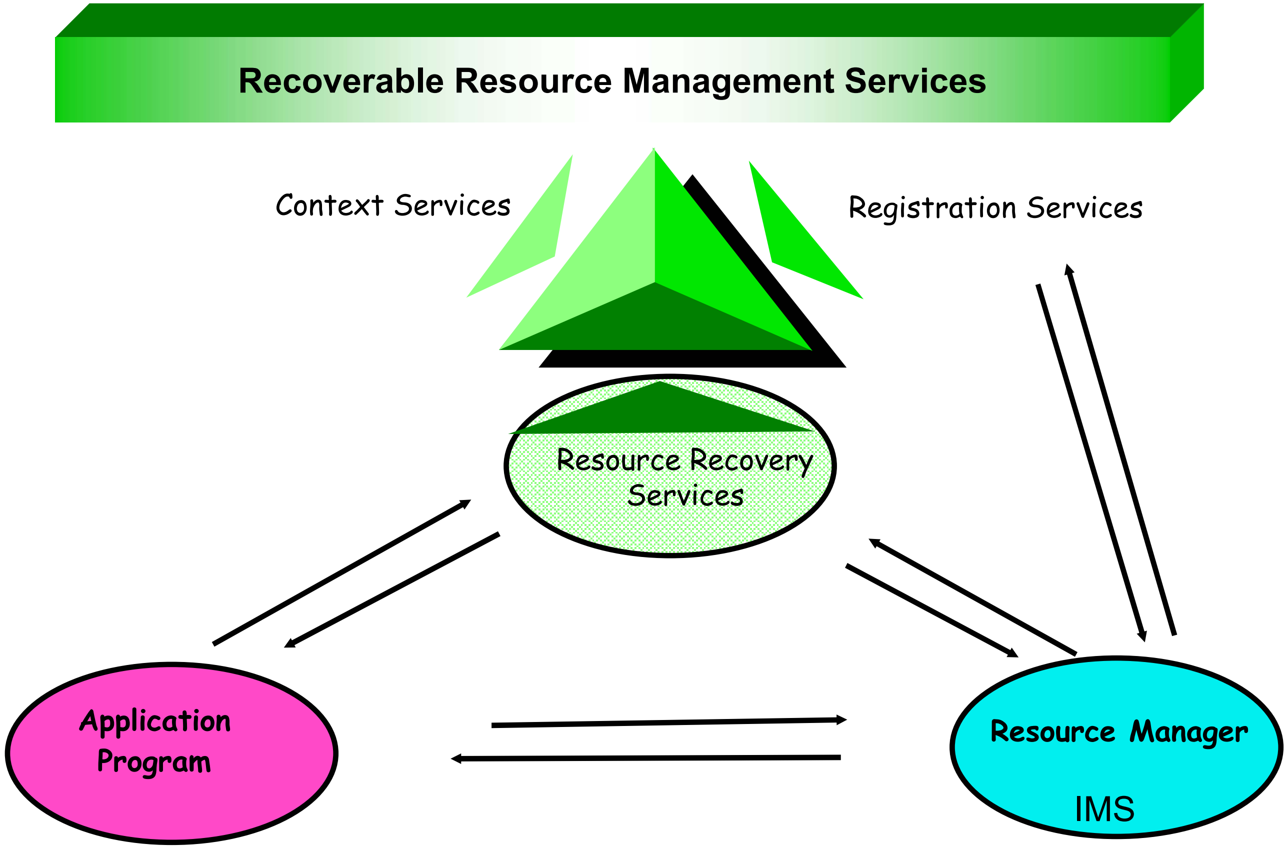
Agenda

- Basic Concepts of z/OS Resource Recovery
- Application related to IMS
  - IMS Batch
  - DB2 Stored Procedure
  - IMS OTMA and APPC Protected Transaction
  - IMS V8 OTMA/APPC Shared Queue exploitation
- RRS Considerations
- Summary

The z/OS Resource Recovery Service (RRS) starts to be widely used in the zOS environment. This session summarizes the different features of IMS which use this new zOS component:
IMS Batch, DB2 Stored Procedures, IMS OTMA and APPC Protected Conversation,
IMS V8 OTMA/APPC Shared Queue.

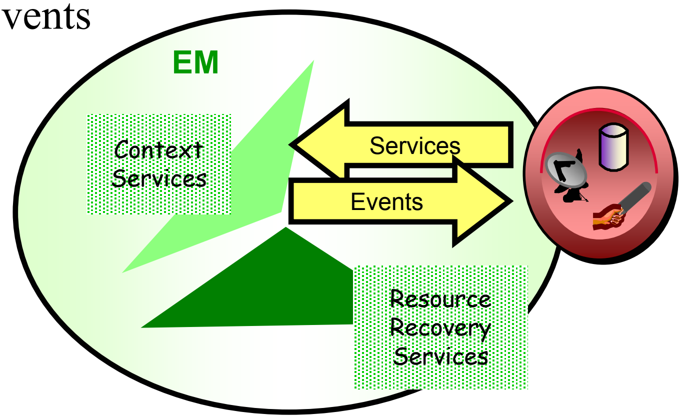
z/OS Resource Recovery

- The z/OS Resource Recovery consists of the protocols and program interfaces that allow an application program to make consistent changes to multiple protected resources.
- Protected resources are sometimes called recoverable resources.

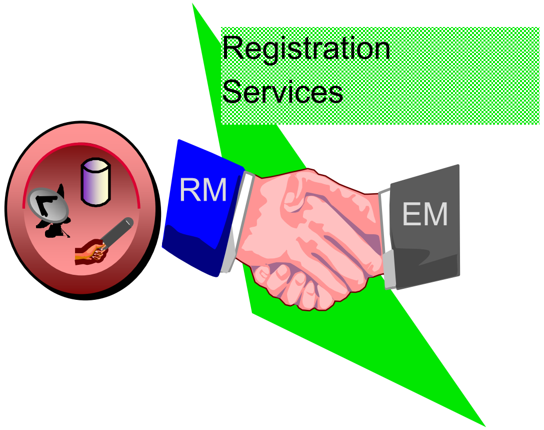


z/OS Resource Recovery - RRM Services

- Exit Manager
  - Authorized program that controls the flow of a predefined set of events
    - Provide Services to RMs
    - Inform RMs of events via exits
  - Two exit managers in z/OS
    - Resource recovery services (RRS)
    - Context services

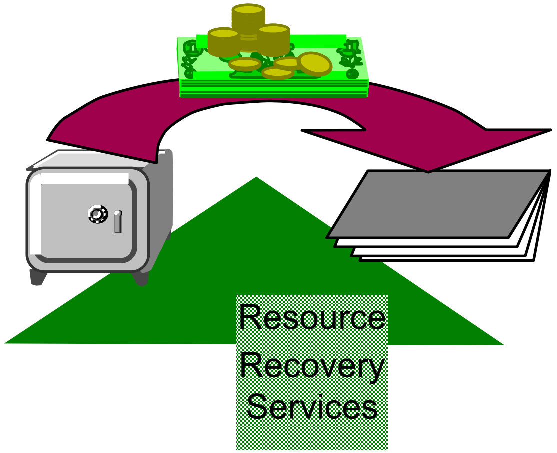


- Registration Services
  - Connecting RMs and EMs
    - Always available
    - Is informed by EMs and RMs when they are available
    - Informs EMs when RMs come and go
    - Informs RMs when EMs come and go
    - Provides global anchors for each

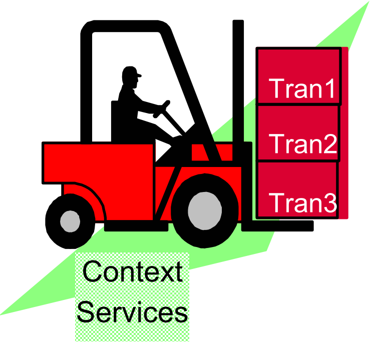


z/OS Resource Recovery - RRM Services …

- Resource Recovery Services - The Syncpoint Manager
  - Coordinates the two-phase commit process
  - Creates an association between a Unit of Recovery (UR) and a Work Context
  - Preserves UR state across all failures
  - 2 Types of Resource Recovery
    - Local on a single system
    - Distributed on multiple systems
      - Communication Resource Manager needed (APPC/MVS)

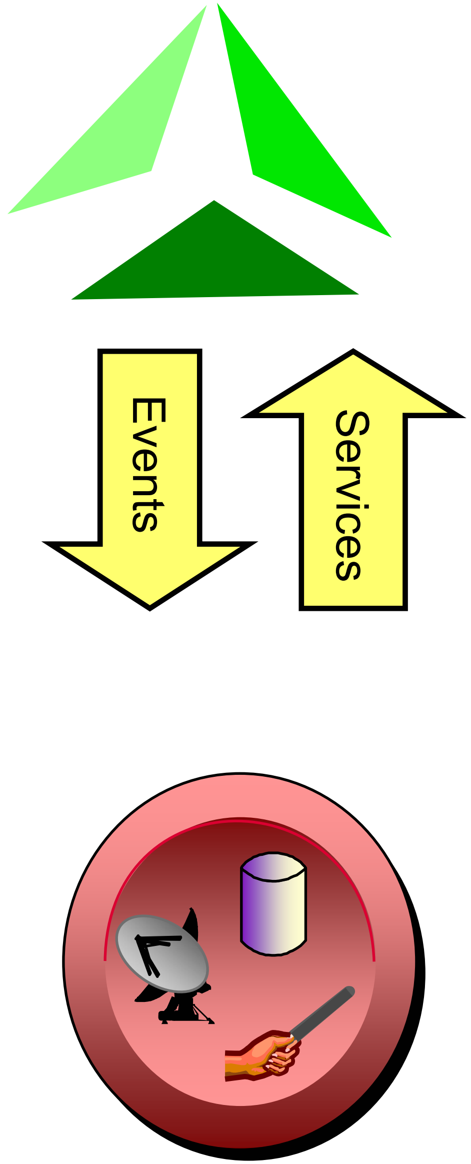


- Context Services - Moving Transactions Around
  - Used by a Work Manager (WM)
  - WM tells CS where the work request is running
  - WMs tells CS when the work request moves
  - RMs are told by CS when the work request is completed



z/OS Resource Managers

- All RMs ...
  - Are authorized
  - Register with the system
  - Supply exit routines
  - Have a system unique name
- Additionally, RRS compliant RMs ...
  - Go through RRS restart processing
  - Have a Sysplex unique name
  - Respond to syncpoint events
- 3 types of Resource Managers
  - Data Resource Managers
    - DB2 DB, IMS DL1, VSAM
  - Communications Resource Managers
    - APPC/Protected Conversations, TRPC, OTMA
  - Work Managers
    - IMS TM, CICS TS, DB2 Stored Procedures, OTMA Client
- 2 types of Resources
  - Protected resources
    - Can be recovered if a failure ocurs
    - Need logging
  - Unprotected resources



z/OS Application Programs

- IMS Batch Application
  - Since IMS V7 with IMS Batch RRS support
    - IMS calls to access IMS databases in the IMS Batch
      - No IMS call limitation
    - SQL execs to access DB2 databases
    - MQI calls to access MQSeries resources

- Any MVS Application connected to RRS
  - Using ODBA for IMS databases access
    - IMS calls to access IMS databases thru an IMS Control Region (DBCTL or DBDC)
      - Only a subset of IMS calls
      - No CHKP/XRST capability
    - SQL execs to access DB2 databases
    - MQI calls to access MQSeries resources
  - For example, DB2 Stored Procedure
  - For example, WAS EJB

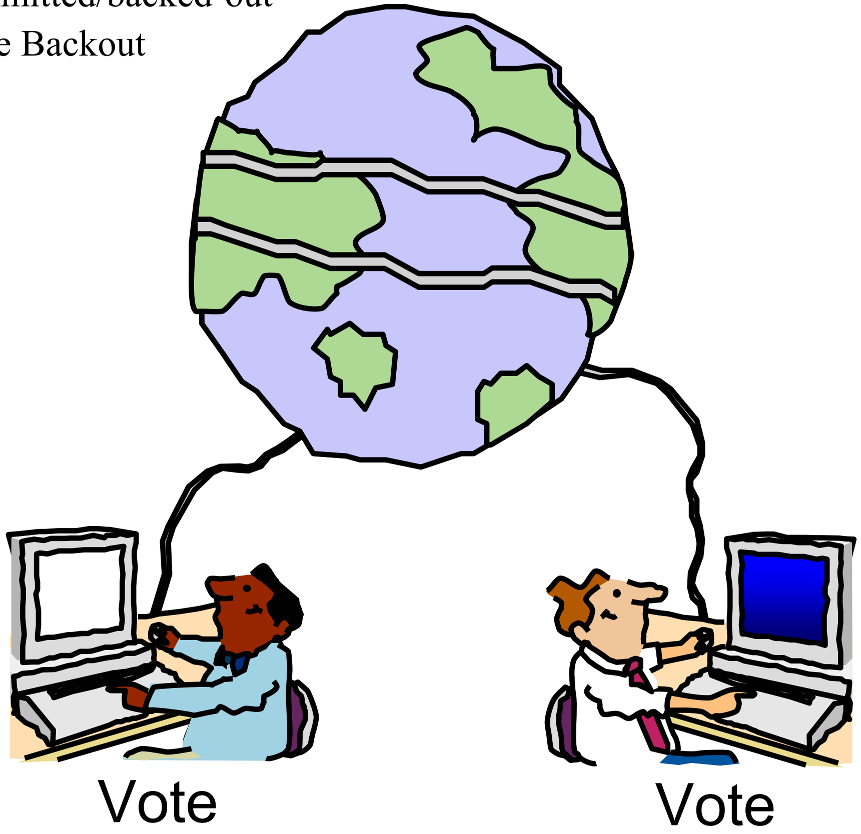
- IMS Transaction
  - Involved in a Protected Conversation
    - With APPC or OTMA
  - In the Shared Q environment with synchronous APPC/OTMA

RRS - Syncpoint Processing

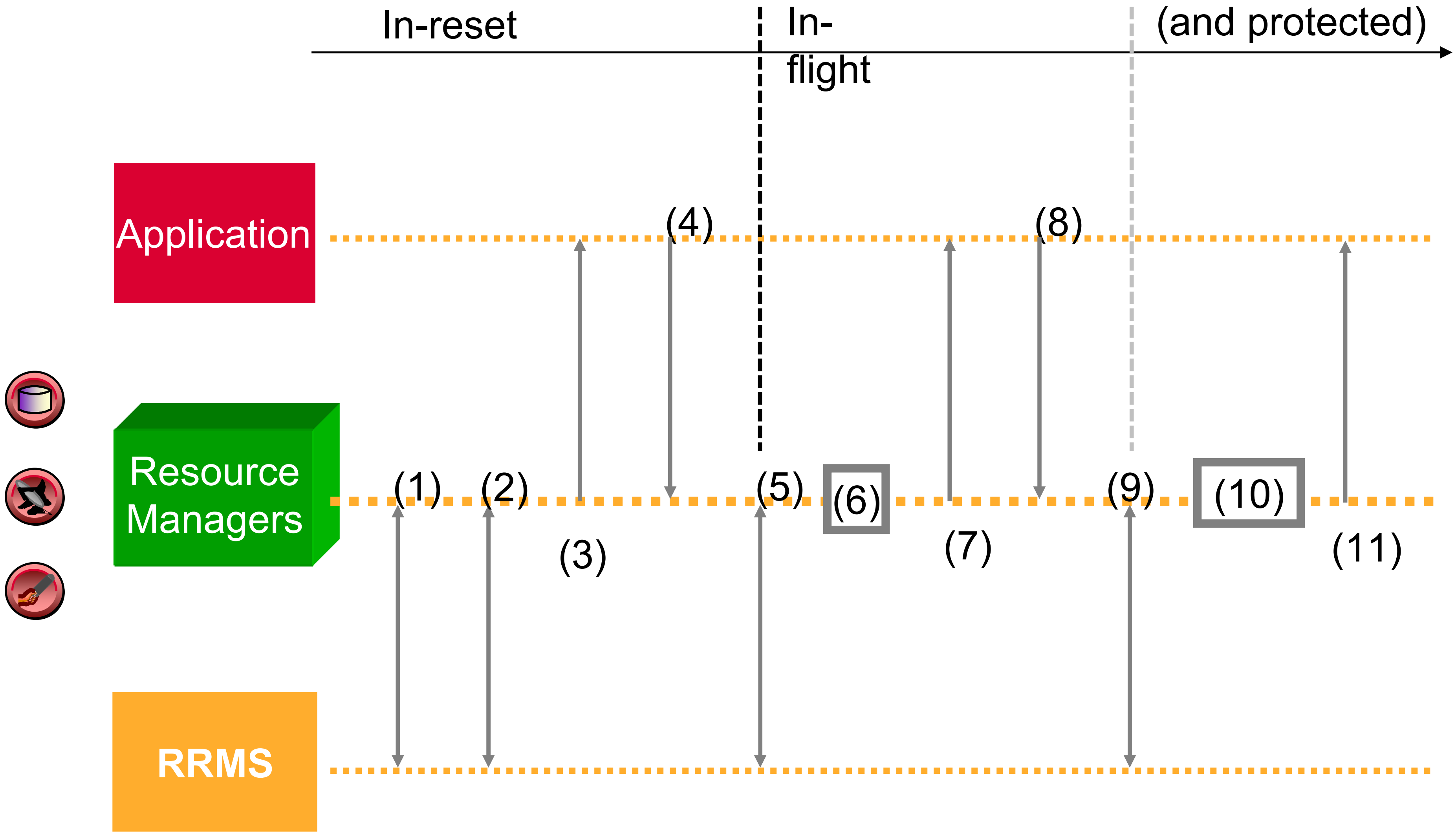
- Sync Point Processing
  - Sync Point Phase 1 - "Prepare" : Each resource manager votes
    - Sync Point Manager polls all Resource managers
    - Each Resource Manager votes to commit or Backout
  - Sync Point Phase 2 - "Commit or Backout" : Changes are committed/backed-out
    - If all votes are to commit then Commit Phase begins, else Backout

- RRS Verbs
  - ATRCMIT or SRRCMIT
  - ATRBACK or SRRBACK

- RRS Unit of Work States
  - IN-FLIGHT :
    - Work changes in process
  - IN-DOUBT
    - Work changes between phase 1 and phase 2
  - IN-COMMIT
    - Work changes are committed
  - IN-BCKOUT
    - Work changes are backed out



Part 1 - The Application Makes Changes

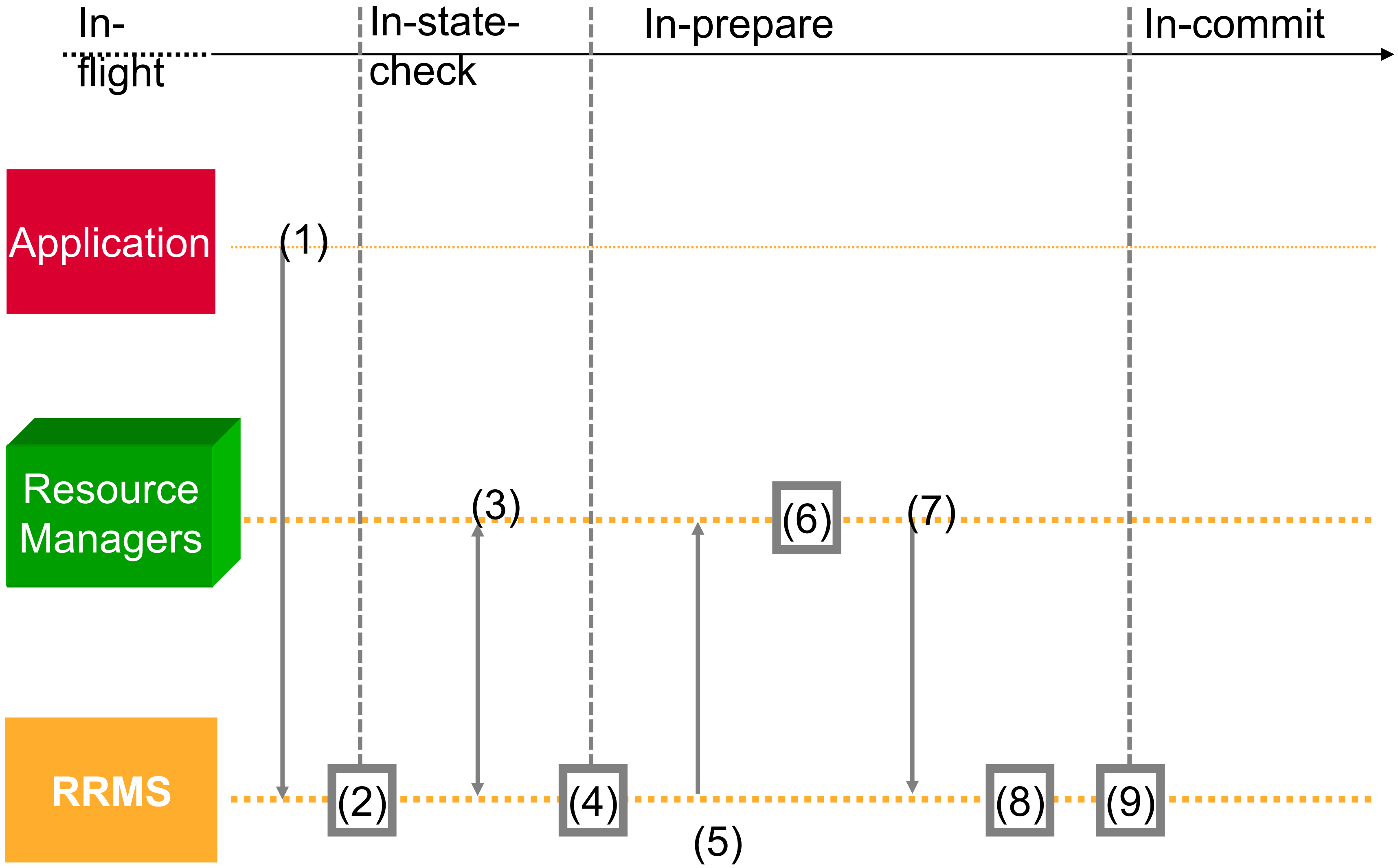


Part 1 - The Application Makes Changes

1. Work Manager gets a private Context from Context Services.
2. Work Manager associates Context with a task (via CS).
3. Work Manager dispatches application.
4. Application makes read request to Data RM1.
5. RM1 expresses unprotected interest in UR, UR and URI1 for RM1 are created.
6. RM1 locks and reads data
7. RM1 returns to application.
8. Application makes write request to Data RM2.
9. RM2 expresses protected interest in UR, URI2 for RM2 is created.
10. RM2 locks data then makes and logs changes.
11. RM2 returns to application

Note: Steps 4,6-7 and 8,10-11 can be repeated.

Part 2 - Syncpoint Processing Begins



Part 2 - Syncpoint Processing Begins

1. Application requests all changes be committed.
2. RRS starts syncpoint process [In-statecheck begins].
3. RRS drives all RM's (optional) statecheck exits and they return.
4. [In-prepare begins] If any RM requested presume nothing protocols, RRS logs a PRP (prepare) record.
5. RRS drives All RM's (required) prepare exits.
6. RMs take prepare actions

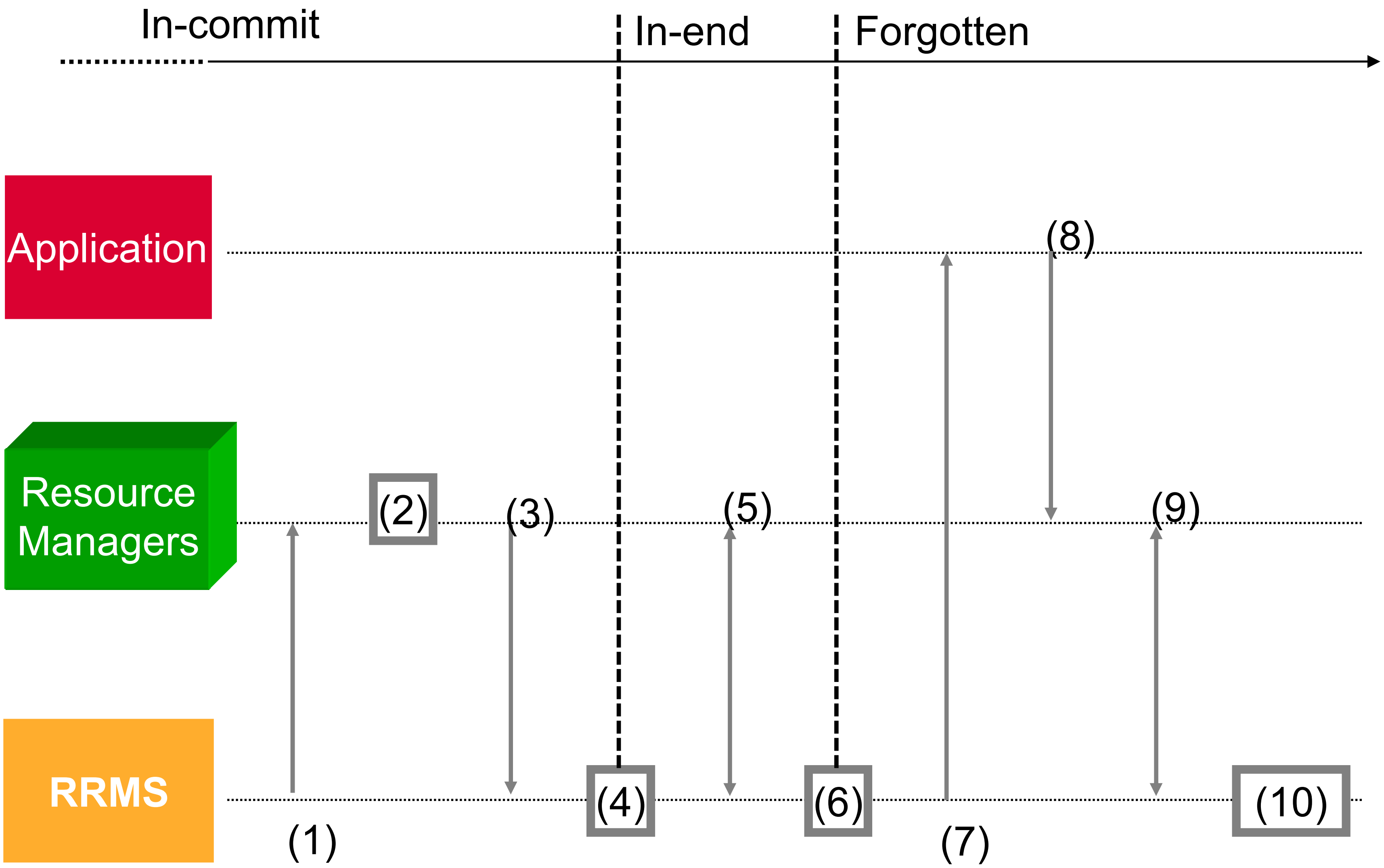
   a. RM1 (read only) releases data locks.

   b. RM2 forces undo/redo records.
7. RMs return from prepare exits

   a. RM1 (read-only) votes forget. It is no longer involved in the UR.

   b. RM2 votes OK.
8. RRS determines overall result (in this case it is OK).
9. RRS logs a CMT (commit) record. This is the "Atomic Instant" [in-commit begins].

Part 3 - Committed and Forgotten

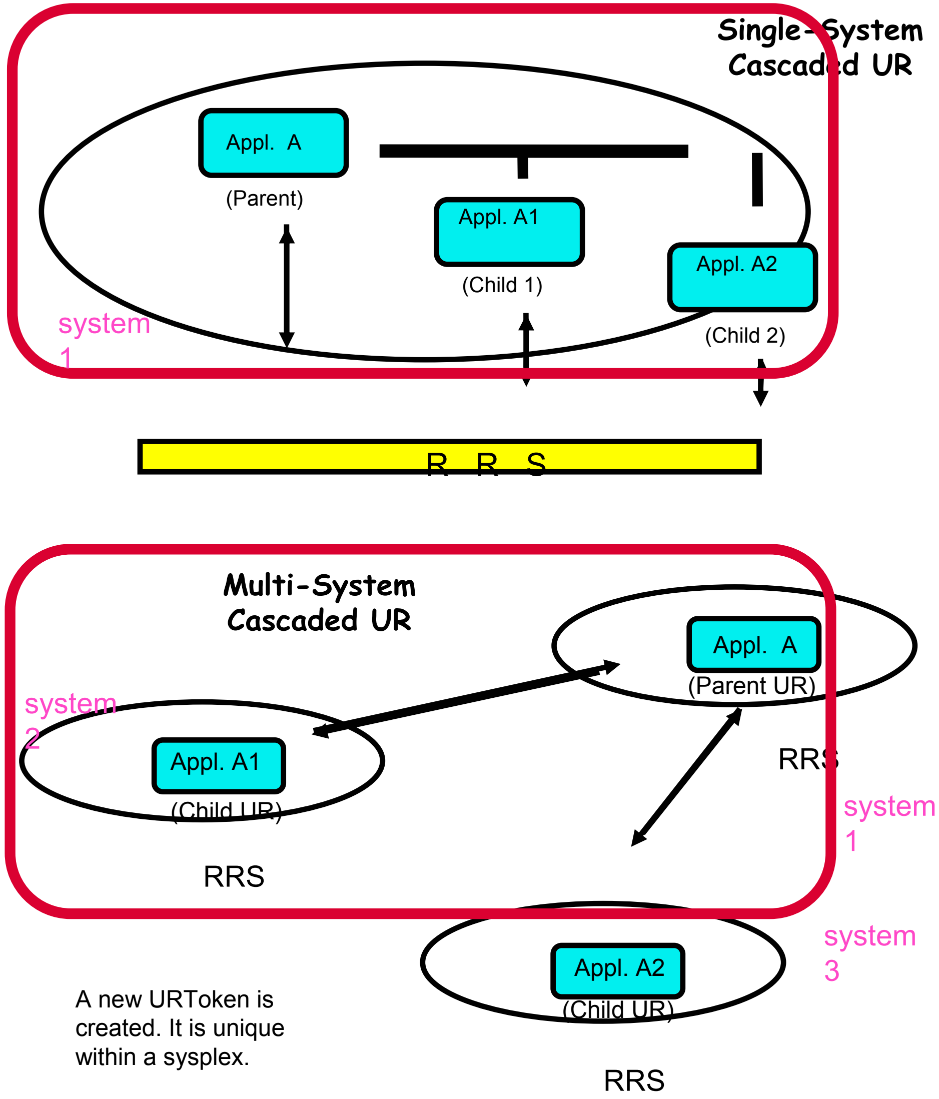


Part 3 - Committed and Forgotten

1. RRS drives RM2's (required) commit exit. (RM1 has forgotten and so its exits are no longer driven).
2. RM2 takes commit action by releasing data locks. The changes are now visible to other work requests.
3. RM2 returns to RRS.
4. RRS begins end processing [in-end begins].
5. RRS drives RM2's (optional) end-UR exit and it returns.
6. UR processing is completed. RRS logically deletes the CMT record from the log.
7. RRS returns to the application informing it that the commit request completed successfully.
8. Application completes and returns to its Work Manager.
9. Work Manager ends the private Context.
10. Sometime later as part of log cleanup, RRS physically deletes the CMT record from the log

RRS Cascaded Transaction

- Created by a work manager when a single work request involves multiple work managers
  - Multiple application parts in parallel
  - All of the changes made by the separate applications can then be part of a single commit scope managed localy by RRS.
- Single-system Cascaded Transaction
  - Introduced in OS/390 V2 R9.
- Multi-system Cascaded Transaction
  - An application executing on one system in a sysplex can have transactional access to data or applications available on another system in a sysplex.
  - RRS coordinates a transaction across multipe systems within a sysplex.
  - The feature was introduced in z/OS V1 R2.



A new URToken is created. It is unique within a sysplex.

Agenda

- Basic Concepts of z/OS Resource Recovery
- Application related to IMS
  - IMS Batch
  - DB2 Stored Procedure
  - IMS OTMA and APPC Protected Transaction
  - IMS V8 OTMA/APPC Shared Queue exploitation
- RRS Considerations
- Summary

RRS Support for IMS Batch

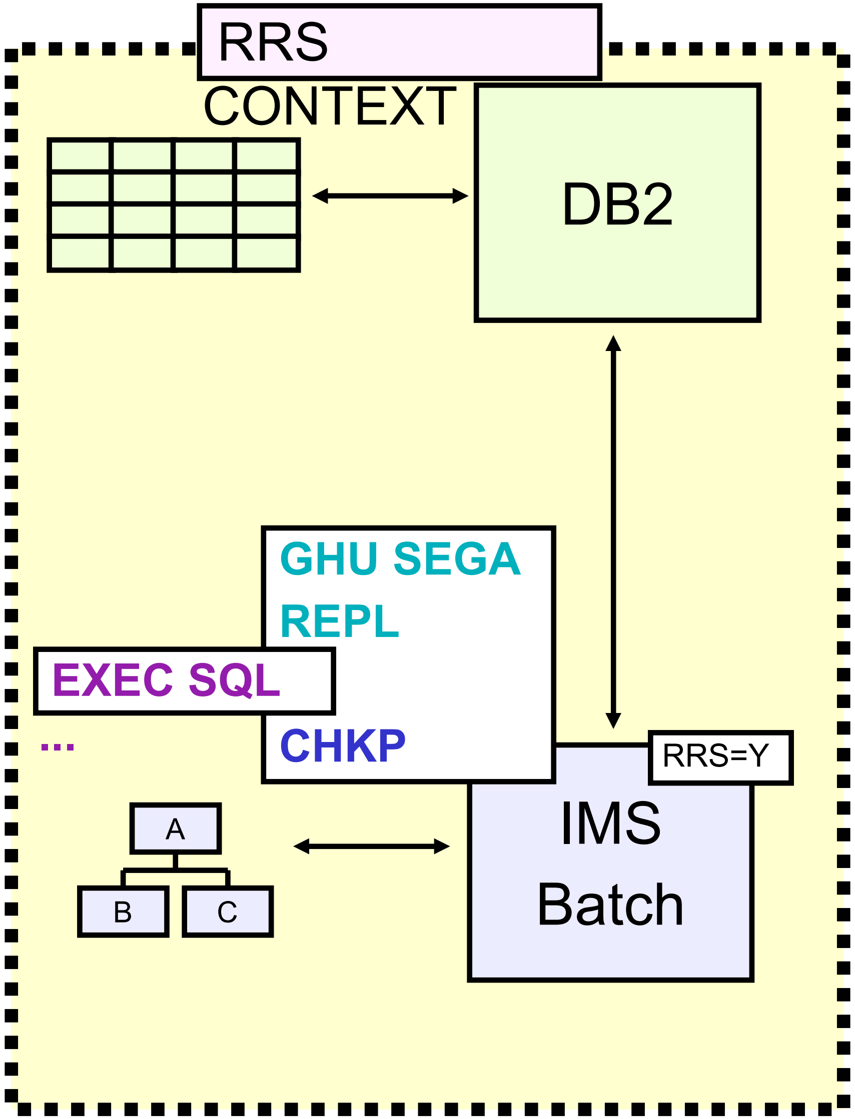
- In a Stand-alone IMS Batch environment
  - IMS V8 now provides full coordinated two phase commit with attached subsystems
    - DB2 V6 or MQ V5.2
  - Batch program requires connection to RRS
    - New JCL PROC parameter “RRS=Y”
    - RRS uses System Logger for saving coordination data
  - Batch program requires DASD logging and BKO=Y

- Used by IMS DataPropagator V3R1
  - With MQ to give “asynchronous near real time” data propagation for Batch DL/1
    -

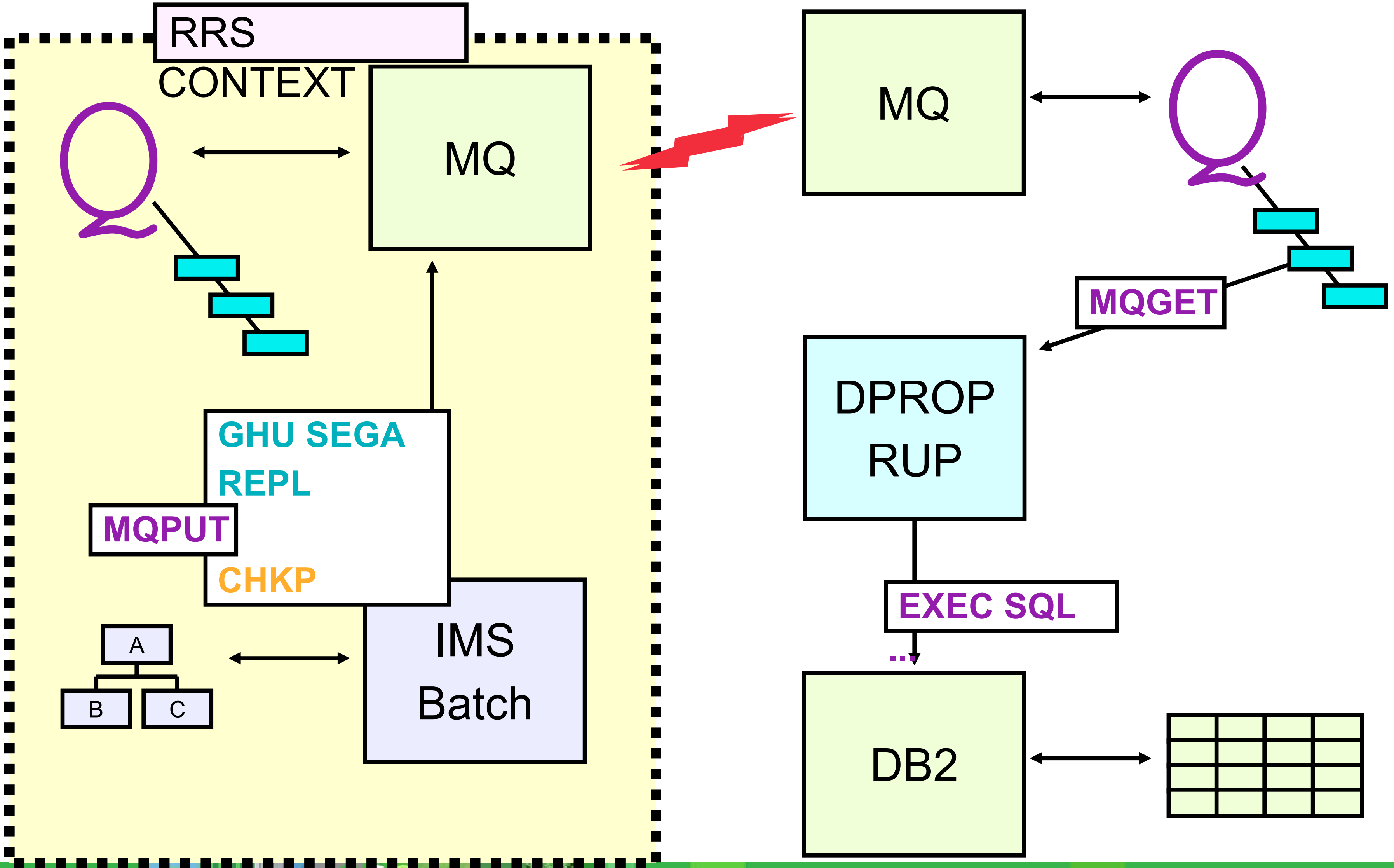
- IMS V7 support with APAR PQ51895

RRS Support for IMS Batch – IMS DProp/DB2

- IMS and DB2 register with RRS
- IMS updates IMS DB
- IMS expresses interest in UR
- IMS DProp Data capture exit updates DB2
- DB2 expresses interest in UR
- CHKP - initiate 2-phase commit for UR
- RRS manages 2-phase commit

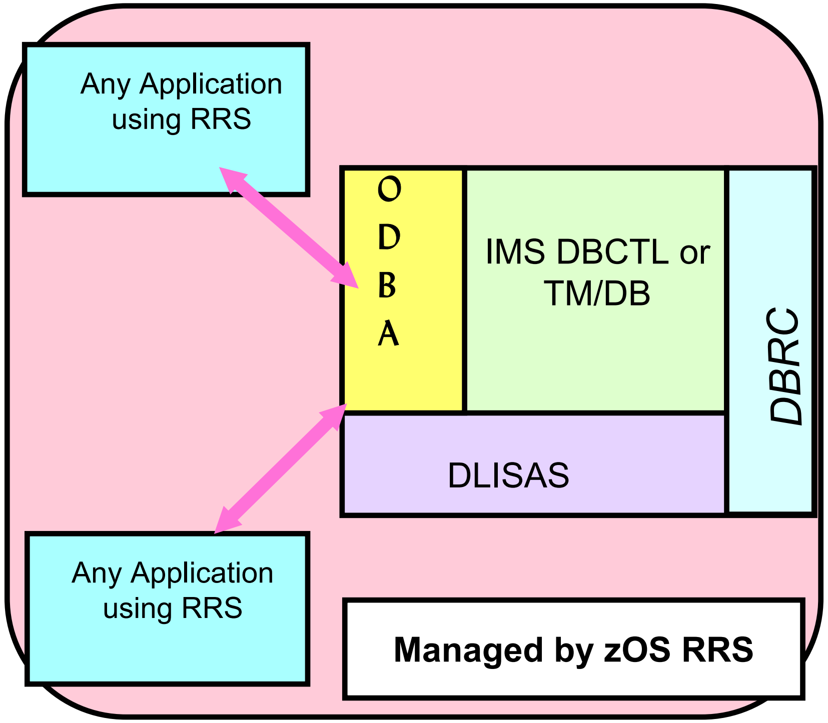


RRS Support for IMS Batch - IMS DProp/MQSeries



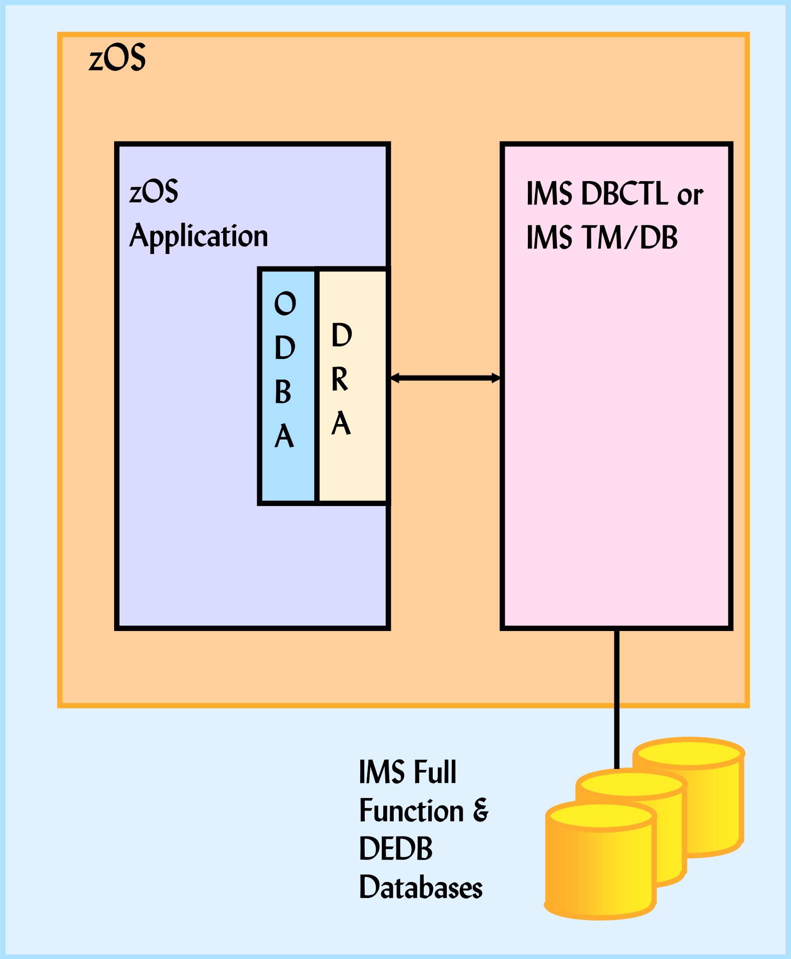
Open Database Access (ODBA) - Objectives

- Provides a callable interface to IMS databases from any z/OS programs that are not managed by IMS
  - DB2 Stored Procedures
  - or Any z/OS applications that use Resource Recovery Services (RRS) to manage their syncpoint processing
- Connection to IMS TM or DBCTL
  - Uses the Database Resource Adapter (DRA)
  - DL/I calls are issued using the Application Interface Block (AIB) interface



ODBA - Database Resource Adapter (DRA)

- The z/OS application gains access to IMS Full Function & DEDB databases through the Database Resource Adapter
  - Similar to CCTL's usage of the DRA
  - ODBA uses the AERTDLI interface rather than PAPL parameter list to communicate its requests to the DRA
- The DRA is initialized as a result of a CIMS INIT or APSB call
- The DRA can process multiple threads at a time
  - Through the MINTHRD & MAXTHRD parameters in the DRA Startup Table
  - Maximum number of threads & dependent regions supported by an IMS DB instance at one time is 999



ODBA's Use of the DRA

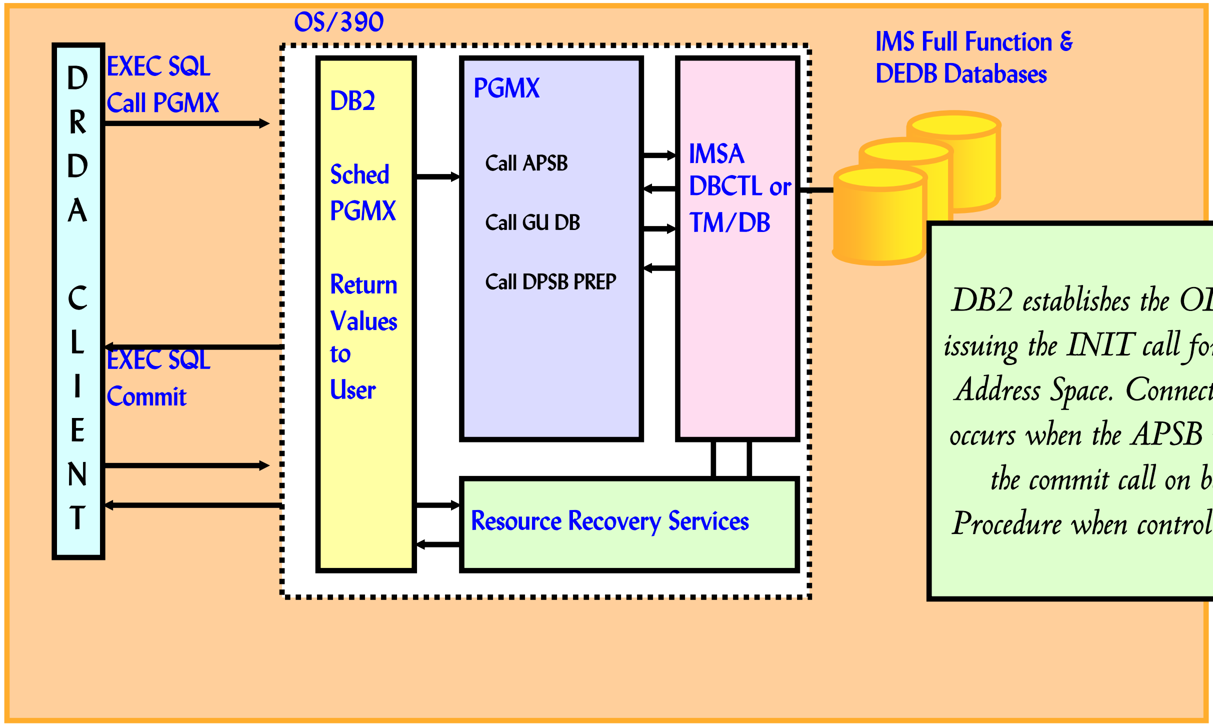
DB2 Stored Procedures

- DB2 Stored Procedures
  - Enables users to "bundle" multiple SQL statements into a single unit
  - A "stored procedure" resides on the mainframe
  - Clients can then execute these statements by issuing one network call
- DB2 - established Stored Procedures
  - Introduced in DB2 V4R1
  - One address space, multiple tasks
  - Limited function -- will not be enhanced
  - Must use DSNALI (call attach)
- WLM - established Stored Procedures
  - With DB2 V5 and OS/390 R3
  - Multiple "environments"
  - Multiple address spaces for each environment
  - Each address space can have multiple tasks
  - Must use DSNRLI (RRS attach)
  - Dynamically started by OS/390 Workload Manager if in GOAL mode

Stored Procedures are blocks of code that contain SQL statements and reside on a server platform.

Access to IMS Databases using call DLI (ODBA)

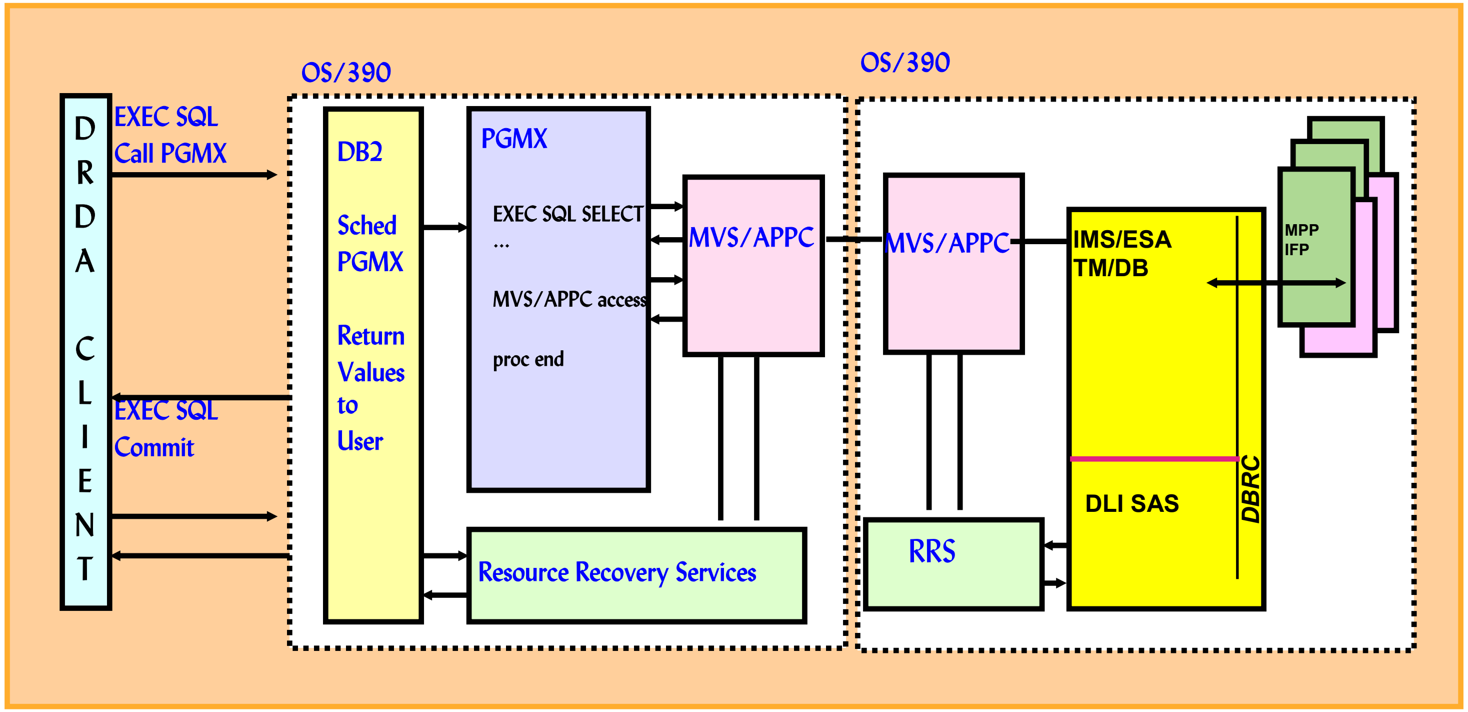
- DB2 Stored Procedures Address Space access to an IMS DB Subsystem
- DL/I data can be presented through an SQL interface either
  - locally to this DB2 or
  - to DRDA connected DB2s
- RRS coordinates the commit between DB2 and IMS



DB2 establishes the ODBA environment by issuing the INIT call for the Stored Procedure Address Space. Connection to a specific IMS occurs when the APSB is issued. DB2 issues the commit call on behalf of the Stored Procedure when control is returned to DB2.

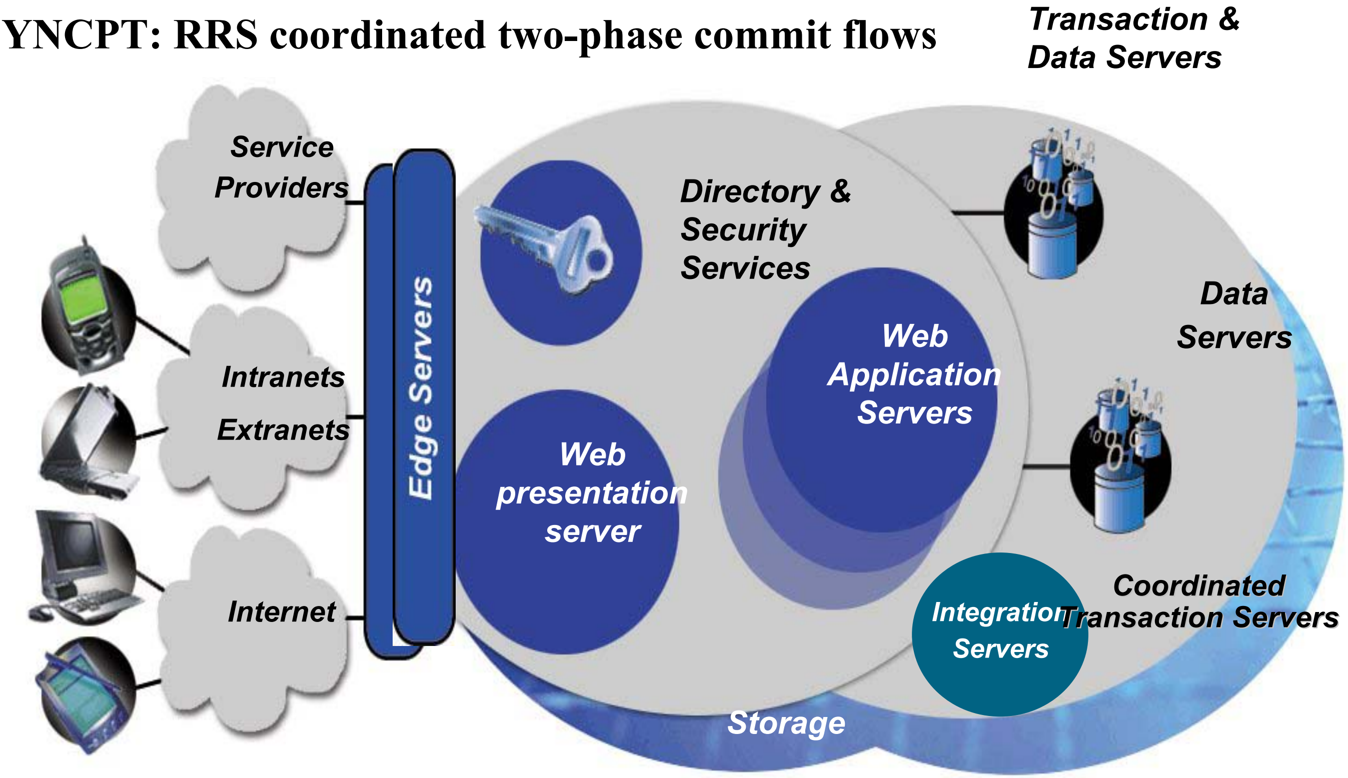
Access to IMS Transactions using APPC

- DB2 Stored Procedures Address Space uses MVS/APPC calls to access IMS TM Subsystem
  - On same OS/390 or on different OS/390
- DB2 notifies RRS about Commit/abord decisions
- RRS passes decision to other resources manager
  - Allows 2-phase commit with MVS/APPC applications (e.g. an IMS Transaction)



Extending the Enterprise with RRS

- IMS OTMA and APPC Protected Transaction
  - OTMA/APPC Synchronization Level (Sync_Level)
    - None : synchronization not required
    - Confirm : confirmation is required before the syncpoint
    - **SYNCPT: RRS coordinated two-phase commit flows**

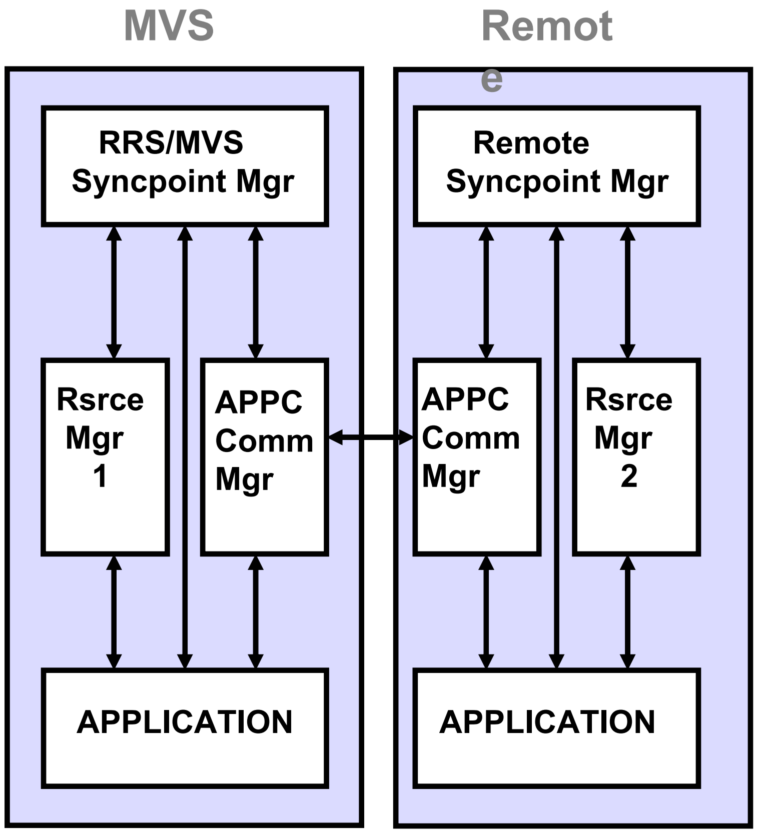


IMS APPC Sync_Level Syncpt Support with RRS

- Distributed Sync Point Support
  - With IMS V6
  - Supported for Standard, Modified Standard, and CPI-C Driven programs
  - Allows IMS applications to be involved in ***protected*** conversations
    - Two phase commit process support provided by RRS
  - Participating platforms/products
    - DB2 V5 and following
    - AIX
    - AS/400
    - VM
    - Windows NT
    - …

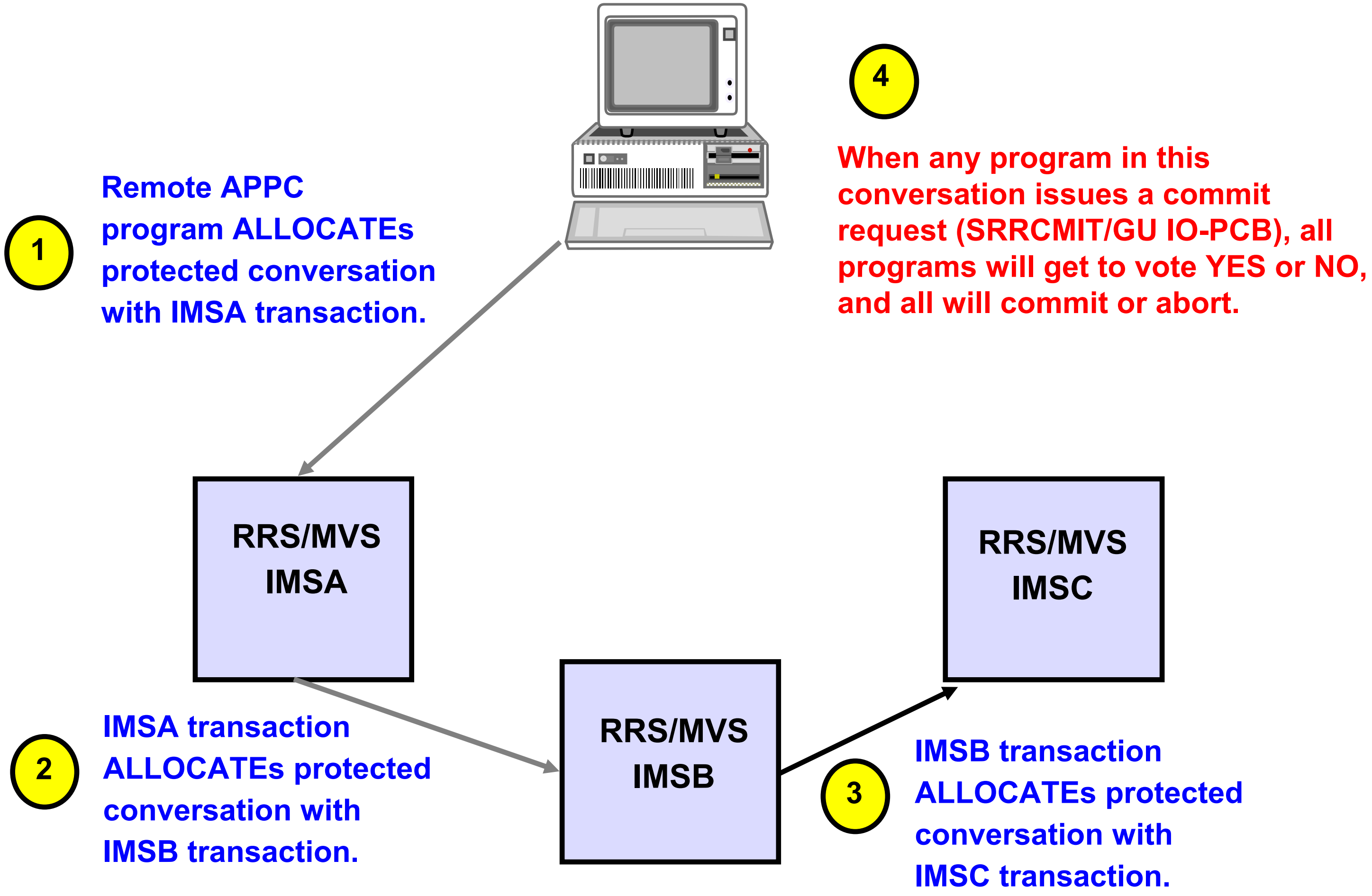
IMS APPC Sync_Level Syncpt Support with RRS …

- Sync Point Manager
  - Coordinates 2-phase commit
  - e.g. RRS/MVS
- APPC Communications Manager
  - Managed communications between components
  - e.g. APPC/MVS
- Resource Manager
  - Controls protected resources
  - e.g. IMS TM/DB
- Application
  - Initiates/participates in protected
  - conversations
  - e.g. IMS MPP



IMS APPC Sync_Level Syncpt Support with RRS …

Example of Protected Conversation



1 Remote APPC program ALLOCATEs protected conversation with IMSA transaction.

4 When any program in this conversation issues a commit request (SRRCMIT/GU IO-PCB), all programs will get to vote YES or NO, and all will commit or abort.

RRS/MVS IMSA

RRS/MVS IMSC

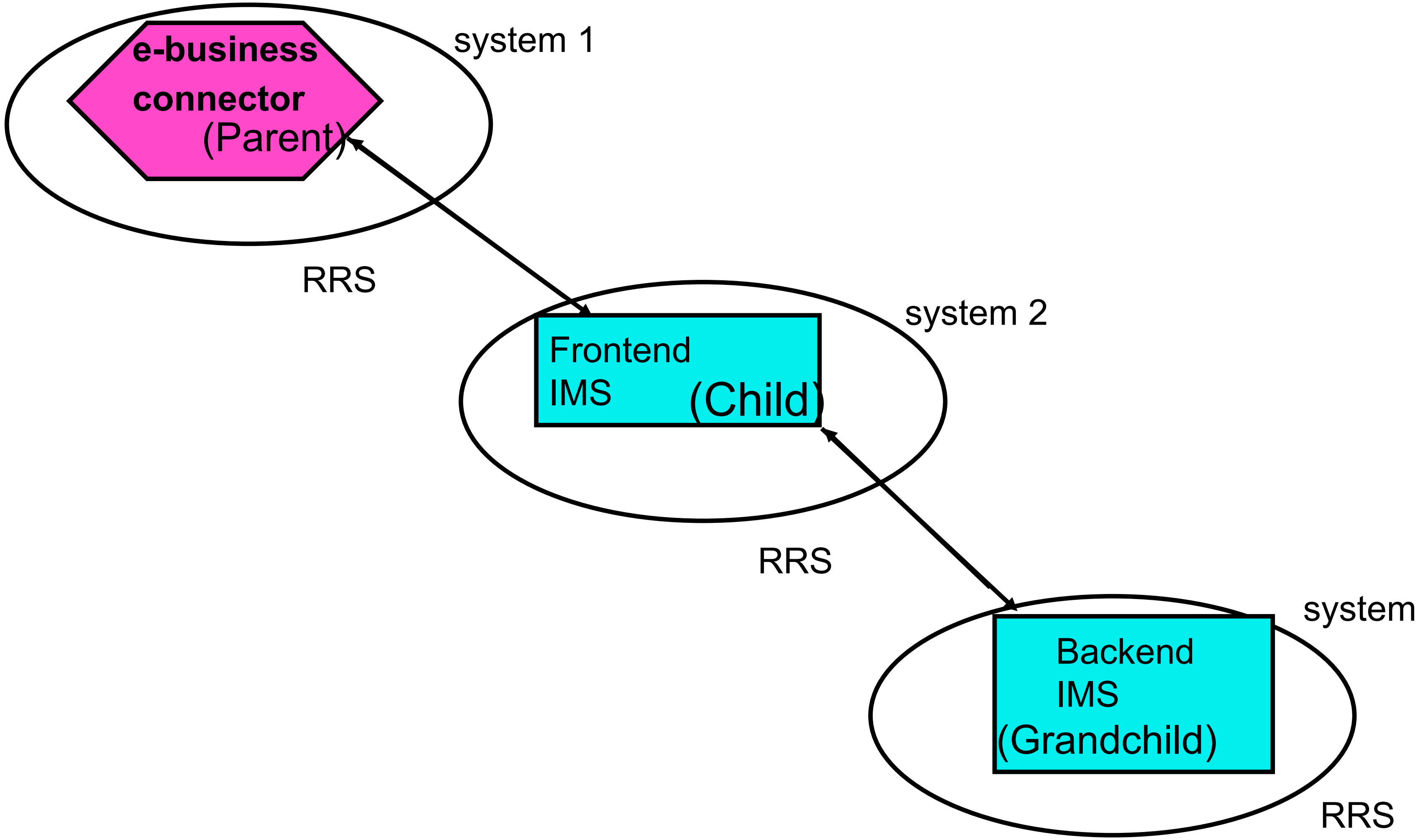
RRS/MVS IMSB

2 IMSA transaction ALLOCATEs protected conversation with IMSB transaction.

3 IMSB transaction ALLOCATEs protected conversation with IMSC transaction.

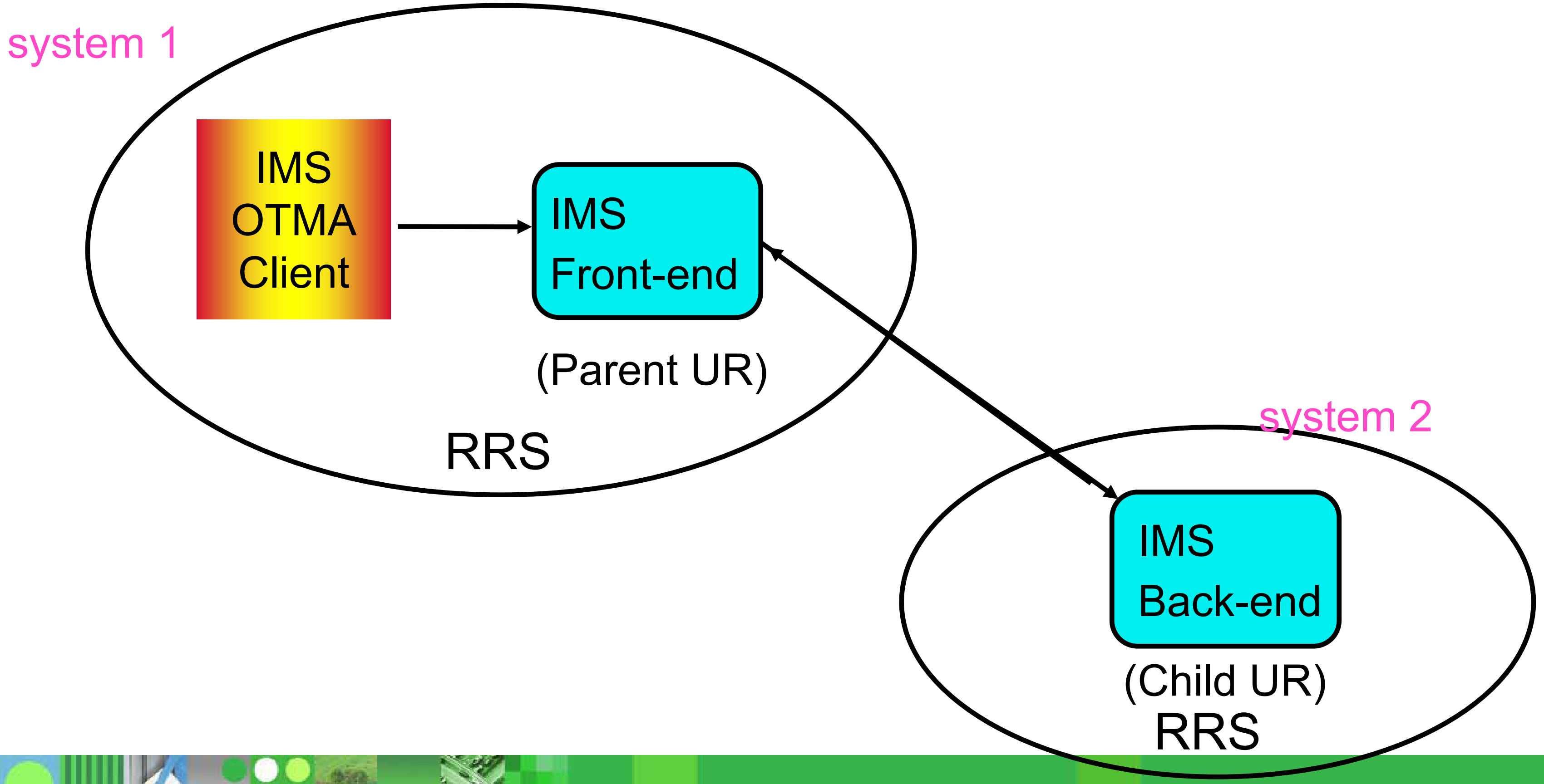
IMS OTMA Sync_Level Syncpt Support with RRS

- **IMS V8 OTMA can process protected transaction (synclevel=syncpt) from different systems.**

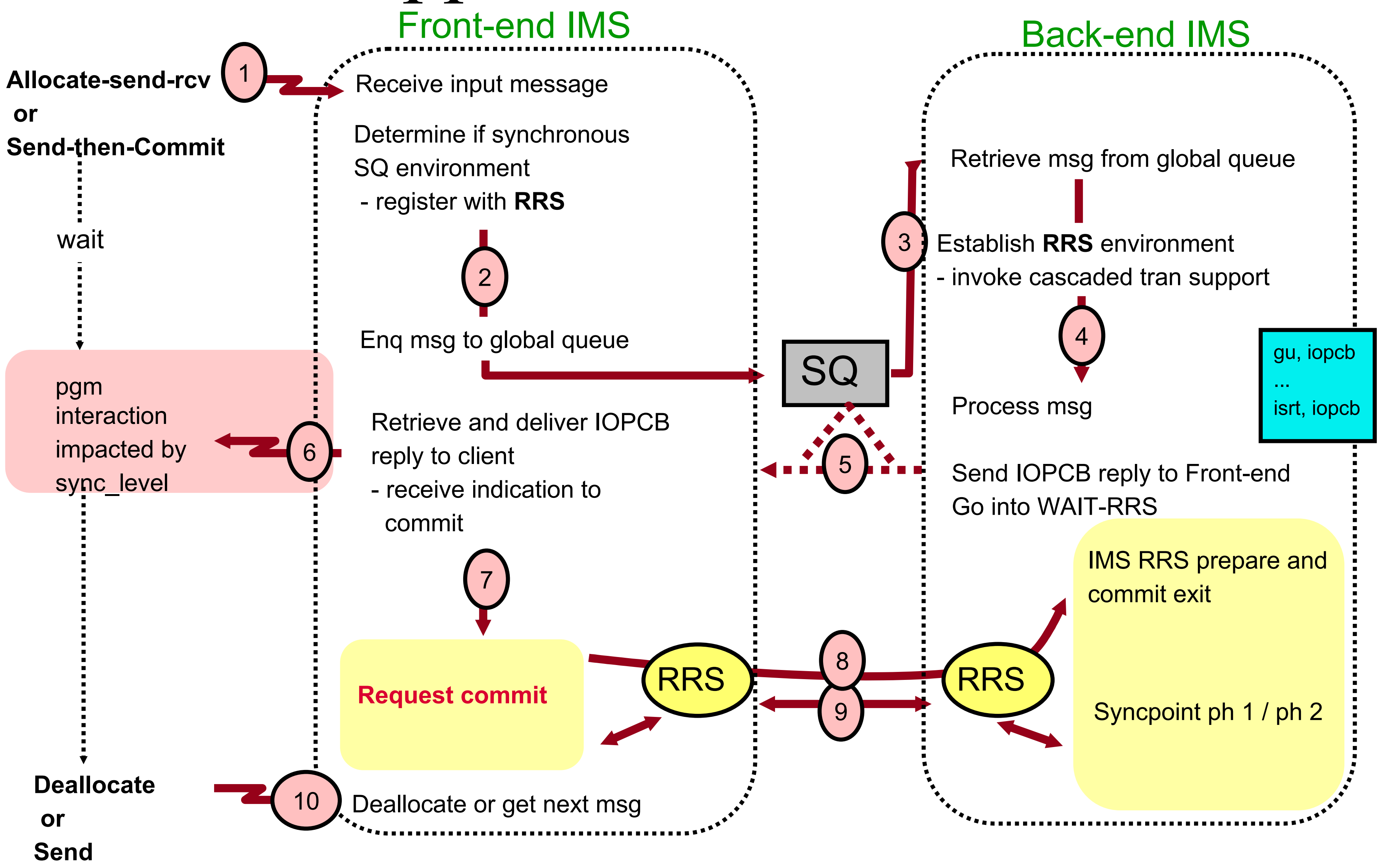


IMS OTMA/APPC Shared Queue Support with RRS

- IMS V8 Synchronous APPC and OTMA Shared Queues
  - Using RRS multisystem cascaded transaction feature.
  - Allows Synchronous APPC/OTMA workload to be distributed and executed on any of the IMS systems in the Shared Queues group.



IMS OTMA/APPC Shared Queue Support with RRS ...



Front-end IMS

Back-end IMS

Allocate-send-rcv
or
Send-then-Commit

1

wait

Receive input message

Determine if synchronous
SQ environment
- register with RRS

2

Enq msg to global queue

SQ

3

Retrieve msg from global queue

Establish RRS environment
- invoke cascaded tran support

4

Process msg

gu, iopcb
...
isrt, iopcb

pgm
interaction
impacted by
sync_level

6

Retrieve and deliver IOPCB
reply to client
- receive indication to
commit

5

Send IOPCB reply to Front-end
Go into WAIT-RRS

7

IMS RRS prepare and
commit exit

Request commit

RRS

8

9

RRS

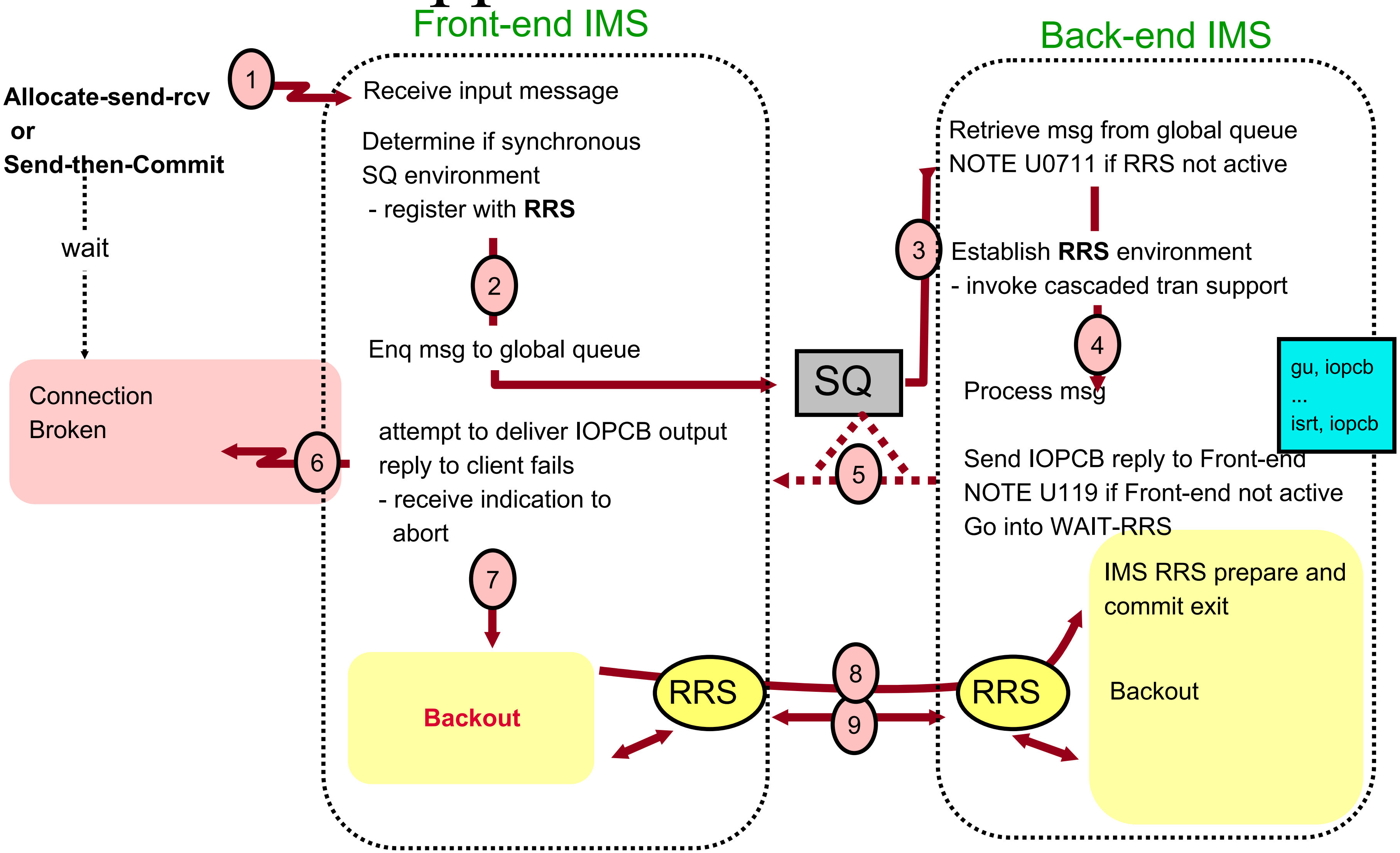
Syncpoint ph 1 / ph 2

Deallocate
or
Send

10

Deallocate or get next msg

IMS OTMA/APPC Shared Queue Support with RRS ...



Front-end IMS

Back-end IMS

Allocate-send-rcv
or
Send-then-Commit

wait

Connection Broken

Receive input message

Determine if synchronous SQ environment
- register with **RRS**

Enq msg to global queue

attempt to deliver IOPCB output reply to client fails
- receive indication to abort

Backout

SQ

Retrieve msg from global queue
NOTE U0711 if RRS not active

Establish **RRS** environment
- invoke cascaded tran support

Process msg

gu, iopcb
...
isrt, iopcb

Send IOPCB reply to Front-end
NOTE U119 if Front-end not active
Go into WAIT-RRS

IMS RRS prepare and commit exit

Backout

Agenda

- Basic Concepts of z/OS Resource Recovery
- Application related to IMS
  - IMS Batch
  - DB2 Stored Procedure
  - IMS OTMA and APPC Protected Transaction
  - IMS V8 OTMA/APPC Shared Queue exploitation
- RRS Considerations
- Summary

RRS Considerations for IMS Customers

- Performance will be impacted by using RRS
- RRS logs many activities related to the RRS events and services.
- The new "RRS=" startup parameter in IMS procedure could be considered to activate/deactivate the RRS function for IMS.

RRS - Installation

- RRS has one XCF group
  - The group has one member for each system in the sysplex
  - The group name is ATRRRS.

- RRS in an MVS system
  - Should be active all the time
  - Should be active on every MVS system image
  - Only one instance of RRS can be active on each system image

- RRS requires system logger
  - an MVS system logger application
    - waits until system logger is active
  - records events related to protected resources
  - five log streams that are shared by systems in a sysplex

RRS - Installation ...

- Dispatching priority of the RRS address space
  - WLM : put RRS in the SYSSTC service class
- Define RRS as a subsystem
  - IEFSSNxx parmlib member :
    - SUBSYS SUBNAME(RRS)
  - does not support dynamic subsystem definition
    - SETSSI ADD,SUBNAME=RRSF
- Starting RRS
  - Sample : in SYS1.SAMPLIB(ATRRRS)
  - Must set-up Proc in SYS1.PROCLIB(RRS)
  - START RRS

RRS - Installation ...

- Stop RRS
  - SETRRS CANCEL
  - if non-resource manager programs in syncpoint
    - may result in a X'058' abend
    - transactions in progress will be resolved when RRS restarts.
  - FORCE RRS,ARM
    - If SETRRS CANCEL does not stop RRS

- Automate RRS handling
  - SETRRS CANCEL command prior to a system IPL
    - result in a cleaner system recovery
  - START RRS command in the COMMNDxx parmlib
    - start RRS during system initialization
  - Restart RRS via ARM if it fails

RRS - 5 log streams

- RRS resource manager data log
  - Info about resource managers using RRS
  - Unconditional duplexing strongly recommended
    - ATR.lgname.RM.DATA
- RRS main UR state log
  - The state of active URs
    - ATR.lgname.MAIN.UR
- RRS delayed UR state log
  - The state of active URs when UR
    - completion is delayed
    - ATR.lgname.DELAYED.UR
- RRS restart log
  - Info about incomplete URs
  - Needed during restart
    - ATR.lgname.RESTART
- RRS archive log
  - Information about completed URs
  - optional (never used by RRS)
    - ATR.lgname.ARCHIVE

RRS - CF Structures

- Sizing
  - CFSizer
    - http://www.ibm.com/servers/eserver/zseries/cfsizer/rrs.html
- Recommendation : 2 structures for all Logstreams
  - Main.UR and Delayed.UR
  - RM.Data and Restart
- Start with no Archive

RRS - Restart

- RRS Warm start :
  - any attempt to start RRS when its logs are not empty
    - also when RRS is started on any system in a sysplex using the same logging group
  - complete work that was in progress when :
    - a previous RRS instance failed
    - or was intentionally stopped

- RRS Cold start
  - when the RRS resource manager data log is empty
  - a sysplex-wide operation
    - affect all RRS subsystems using the logging group
  - cannot complete any work that was in progress

- IXCMIAPU utility to delete and redefine the RRS RM data log
  - sample in ATRCOLD member of SYS1.SAMPLIB

RRS - ISPF Panels

- Enables view of :
  - RRS logs
  - UR information
  - Resource manager information

- Enables decisisions :
  - Determine where a resource manager can restart after a system failure
  - Resolve an in-doubt state for a UR to in-commit or in-backout
  - Remove a resource manager's interest in a UR

- Enables some 'follow-on' (tuning help)

Summary

- z/OS Resource Recovery allows an application program to make consistent changes to multiple protected resources.
  - IMS batch application programs support 2-phase commit syncpoint processing using RRS.
  - ODBA implementation is possible because of RRS. DB2 Stored Procedure can benefit of it.
  - OTMA/APPC Protected Conversation supported by using RRS.
  - IMS V8 OTMA/APPC synchronous messages are allowed to be processed by any backend IMS using RRS.

References

z/OS V1R4.0 MVS Programming: Resource Recovery

z/OS V1R4.0 MVS Setting Up a Sysplex