

Unit 8 – IMS Data Sharing, Shared Queues and the Common Services Layer

What this unit is about:

Up to this point, we have discussed IMS configured as a standalone system that processes transactions and accesses databases. While this configuration is still widespread, it is becoming increasingly common to have multiple IMS systems working together as an *IMSplex*. In this unit, we will discuss how Data Sharing and the Common Services Layer (CSL) provide for better management of multiple IMS systems. We will also discuss how the CSL supports the Dynamic Resource Definition (DRD) ability to define application resources to IMS without the need for an IMS Gen.

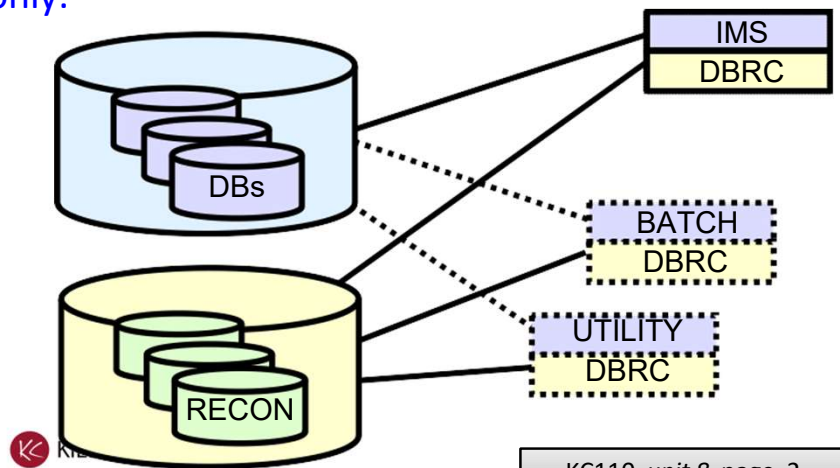
What you should be able to do

After completing this unit, you should be able to:

- Describe the functions and capabilities of IMS Data Sharing
- Understand how processing can be shared between multiple IMS systems
- Identify the additional IMS components introduced by the Common Services Layer (CSL) and how these components assist in the manageability of an IMSplex
- Describe how the Dynamic Resource Definition Facility can be used to add or modify resources defined to IMS systems

IMS before Data Sharing

- Before IMS/VS V1 R2:
 - No data sharing:
 - Only one IMS at a time could access the data
 - This limited capacity to what would run on the largest processor available
 - Databases protected by user (DISP=OLD)
 - To process database in batch or utility region
 - /DBR database - PROCESS - /START database online
 - DBRC used for recovery only:
 - No authorization processing



KC110 unit 8 page 2

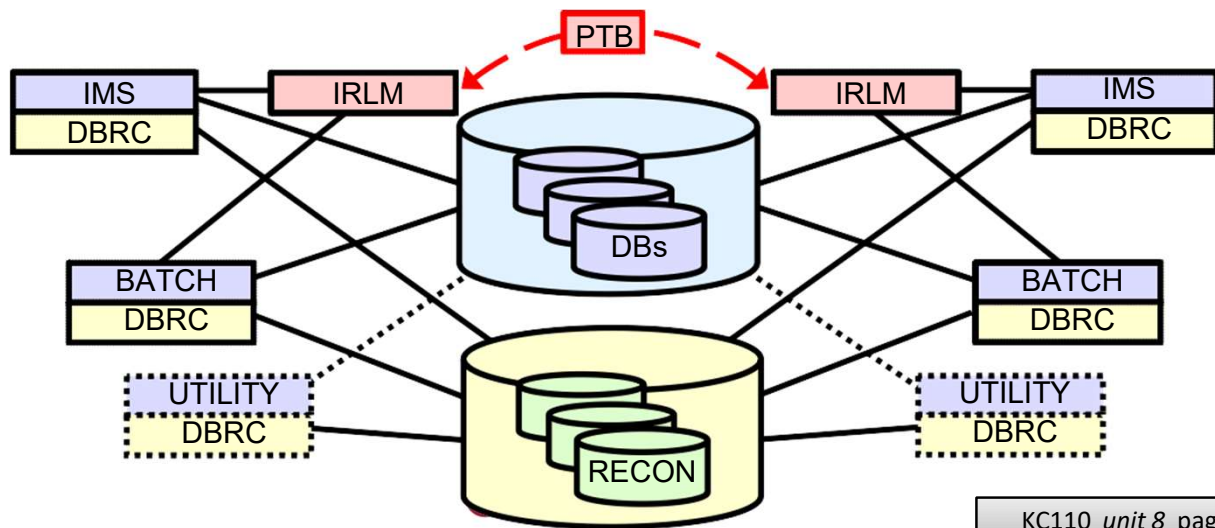
Notes:

IMS V1 R2 introduced IMS-managed data sharing. Prior to IMS 1.2, data could only be shared **WITHOUT INTEGRITY** by coding DISP=SHR on the DBDS DD statement.

To process a database (safely) in another IMS (such as an IMS batch region), it was necessary to DBR the database from the online system, process it in batch, and then restart it in the online region.

DataSharing introduced in IMS/VS 1.2

- Block-level data sharing introduced:
 - DBRC added database authorization processing
 - IRLM added as global lock manager (maximum of two IRLMs)
 - IMS used IRLM for lock management and buffer invalidation
 - IRLMs did communicate using Pass-the-Buck processing
 - PTB processing required significant overhead for lock requests and Buffer Invalidate requests



KC110 unit 8 page 3

Notes:

Version 1.2 brought data sharing with integrity. DBRC assumed responsibility for database authorization processing, which meant that even with DISP=SHR, if DBRC did not indicate that a registered database could be shared, then authorization processing would fail. Global lock management became a function of a new address space called the IRLM, which at that time meant IMS Resource Lock Manager.

Data being shared across multiple MVS images, the IRLMs that times granted locks and coordinated lock management with each other using a process called Pass-the-Buck (PTB) processing by means of a VTAM link between the two IRLMs.

Passed in this Buck Token were Lock Requests, Responses to the requests, and Buffer Invalidate requests. Maximum of 2 IRLMs allowed only, although one IRLM could support multiple IMSs (same LPAR). Many installations did simply share data between an IMS online system and IMS batch jobs (or between CICS local DLI and IMS batch.)

IRLM

- IRLM is required to enable IMS Data Sharing:
 - Since PI ENQ/DEQ tables exist in the DLISAS, sharing of lock information between IMS systems is not possible
 - When IRLM is used, lock tokens are in IRLM EPA
 - When data sharing is performed between processors, lock information is stored and commonly accessed from a Coupling Facility Lock Structure
- IRLM can be used in place of PI as IMS's Lock manager
 - This choice is made by specifying "IRLM=Y" and an irlmname on the *IRLMNM*= IMS startup parameters
 - There are no other parameters that control IRLM in the IMS startup JCL
- IRLM uses Detection instead of Prevention to address deadlocks
 - In the IRLM Startup, a Deadlock Detection Time is specified
- Non-Data Sharing IMS systems can use IRLM in place of PI:
 - However in most cases, PI uses less CPU than IRLM
 - Granularity of locking with IRLM is slightly different than with PI
 - PIMAX= should not be 0, even when IRLM is used
 - PI is still used for some HD Space Management functions

Notes:

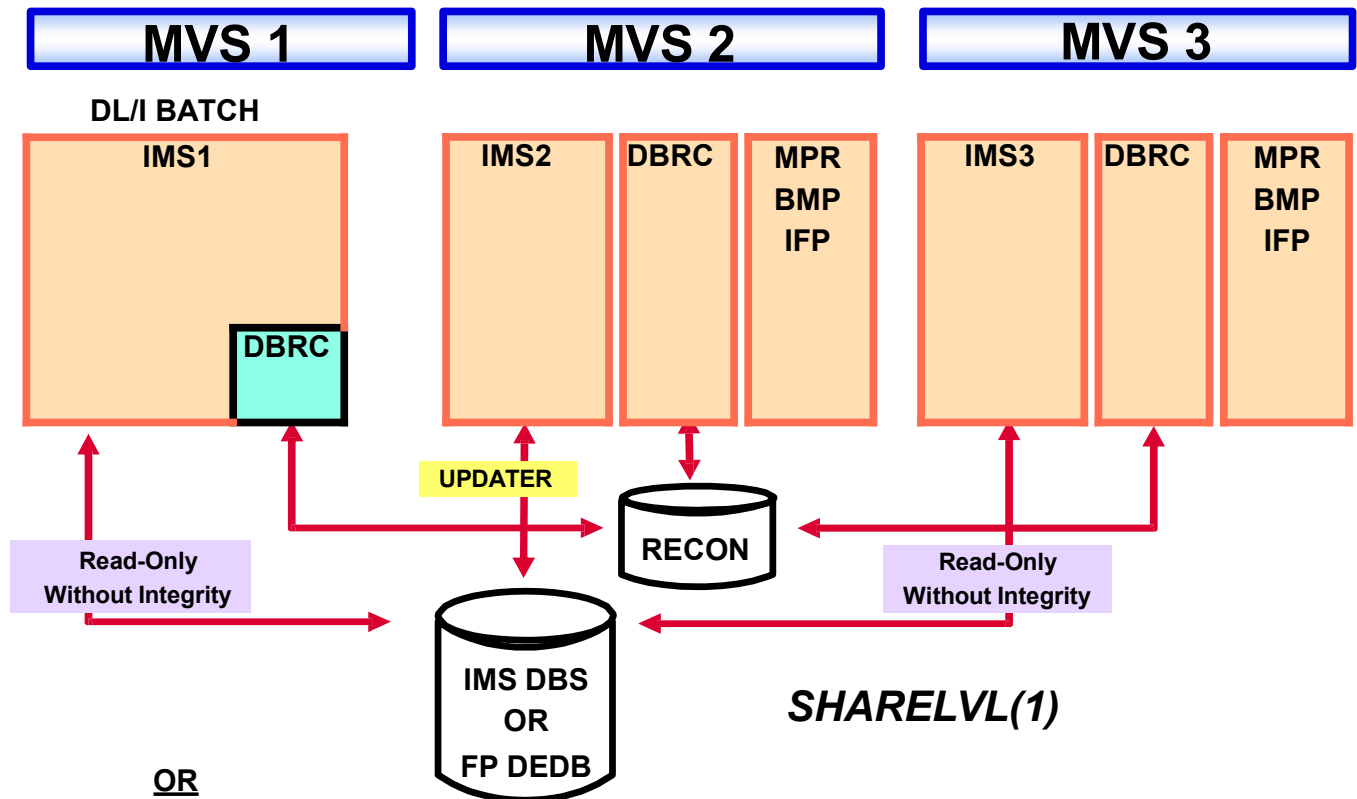
The location of the lock tokens has changed through the different releases of IRLM; In IRLM 2.1 the lock tokens could be stored in either ECSA or EPA. For IRLM 2.2, the lock tokens can only be stored in IRLM EPA,
Current version is 2.3 – for IMS no difference to version 2.2 . IRLM is covered in detail in IMS Block Level Data Sharing class (CM50xx).

DBRC Share-Level Options

- IMS/VS 1.2 (and all versions since) require that a database be *registered* to DBRC in order to be shared between IMSs
 - A database's *SHARELVL* controls its ability to be shared
- Database-level Sharing:
 - *SHARELVL(0)*
 - Guarantees EXCLUSIVE use
 - *SHARELVL(1)*
 - One updater; multiple Read-Only (RO)
 - Multiple readers, RO and/or Read w/Integrity (RD)
- Block-level Data Sharing:
 - *SHARELVL(2)*
 - Multiple UP/RD/RO
 - Maximum one MVS system image
 - INTRA-system, DBRC, one IRLM
 - *SHARELVL(3)*
 - Multiple UP/RD/RO
 - Multiple MVS system images
 - INTER-system, two IRLMs, VTAM *connection*

From the time that Data Sharing was introduced, up through the most current IMS version, in order to share a database with integrity, it must be registered to DBRC and a non-zero value must be specified for SHARELVL; SHARELVL=0 is the default. If a database is registered to DBRC, as part of IMS database open processing, DBRC Authorization is performed; as part of authorization, DBRC checks SHARELVL of the database and the current degree of sharing (for example, is there already an exclusive updater?). If the new IMS authorization request is not compatible, the authorization, and therefore the open process, is rejected.

Database Level Sharing: SHARELVL(1)



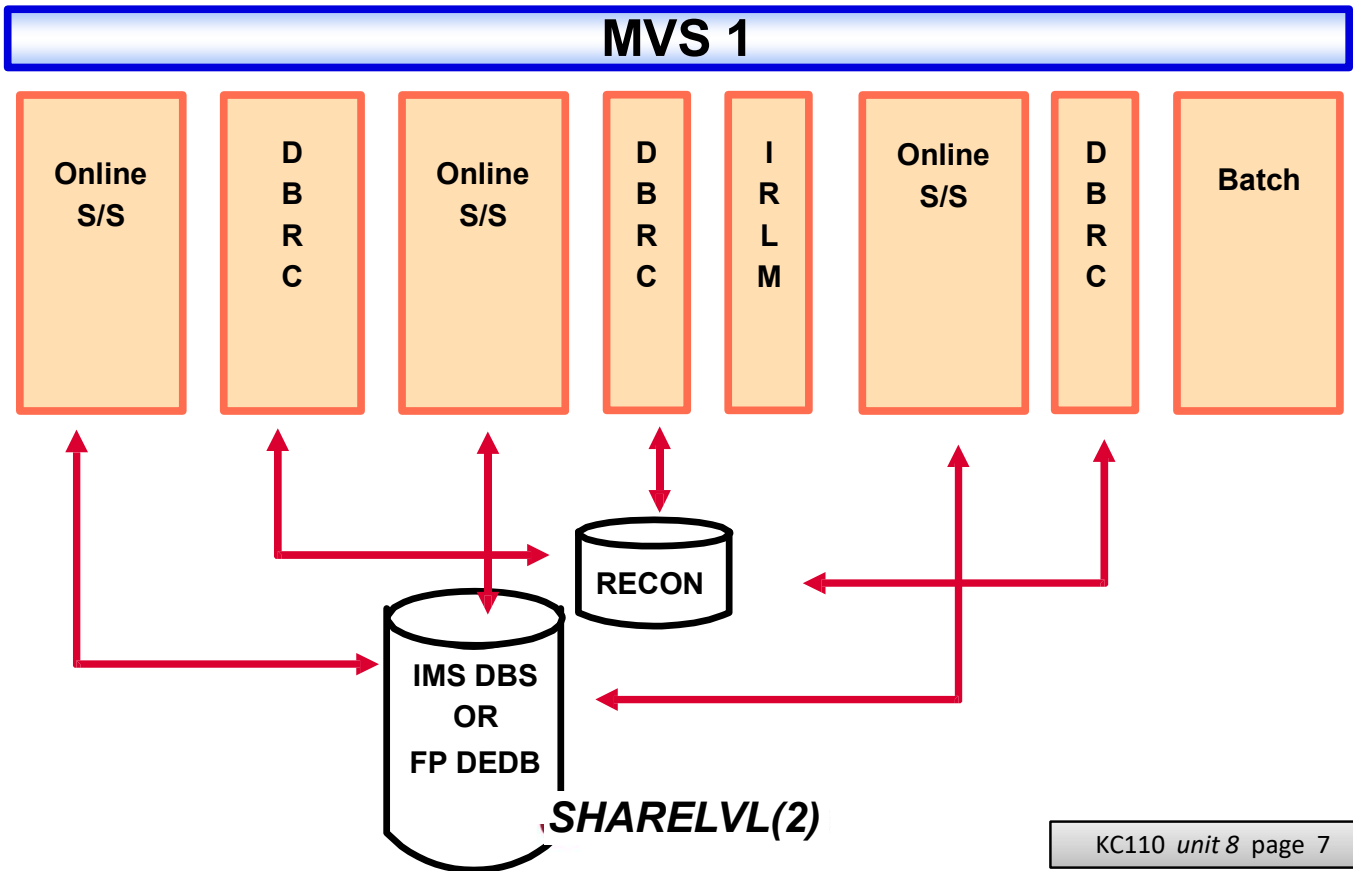
- All subsystems can *READ WITH INTEGRITY*
 - With no updaters, there is no integrity exposure

Notes:

This illustrates the impact of using SHARELVL=1. This is barely sharing since although integrity is assured, there is no additional update capacity. This definition for database-level sharing (and block-level sharing on the next page) has not changed since the original IMS/VS 1.2 implementation.

Block Level Data Sharing *Intra*-System: SHARELVL(2)

ONE MVS SYSTEM:

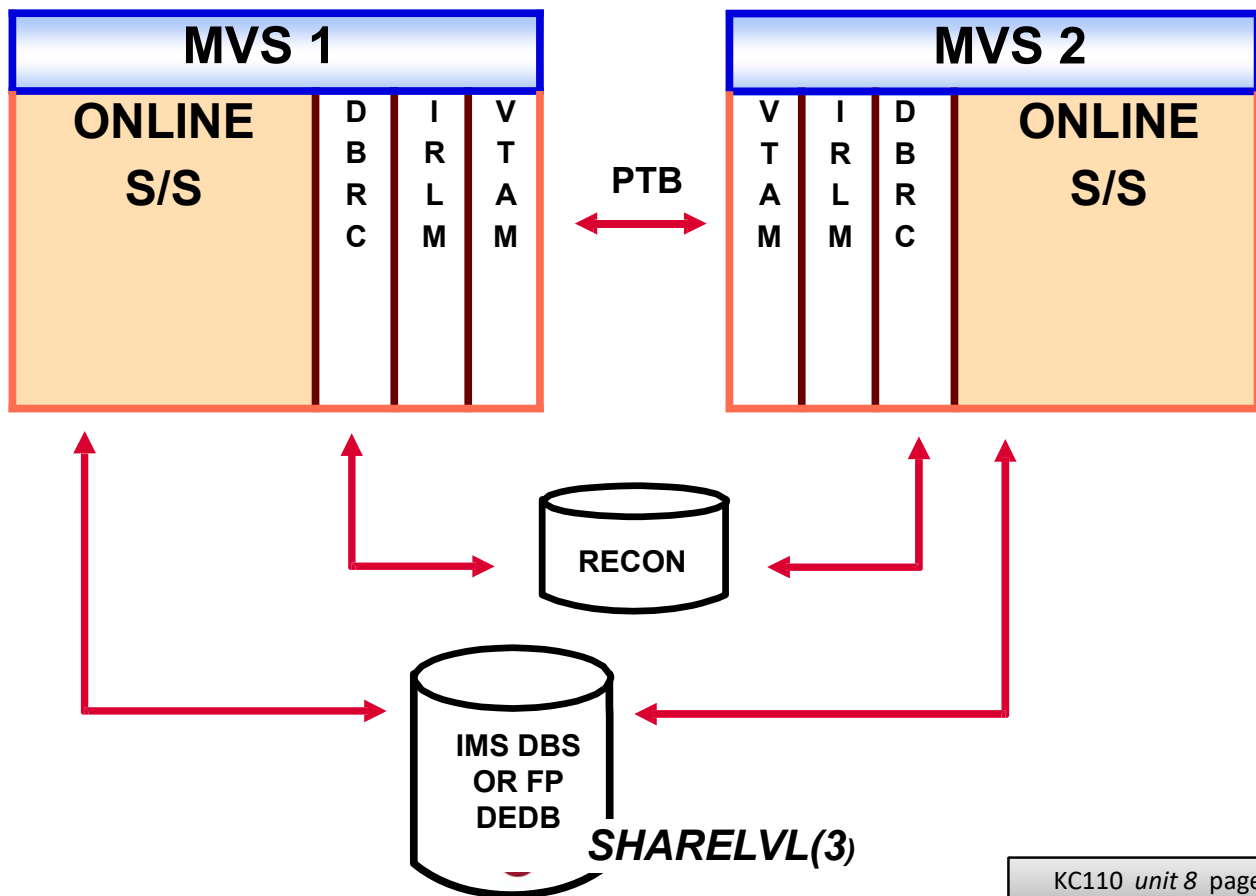


KC110 unit 8 page 7

Notes:

With Share Level 2, there is still no additional capacity, but there is at least some improved flexibility for accessing IMS databases

Block Level Data Sharing *Inter*-System: SHARELVL(3)



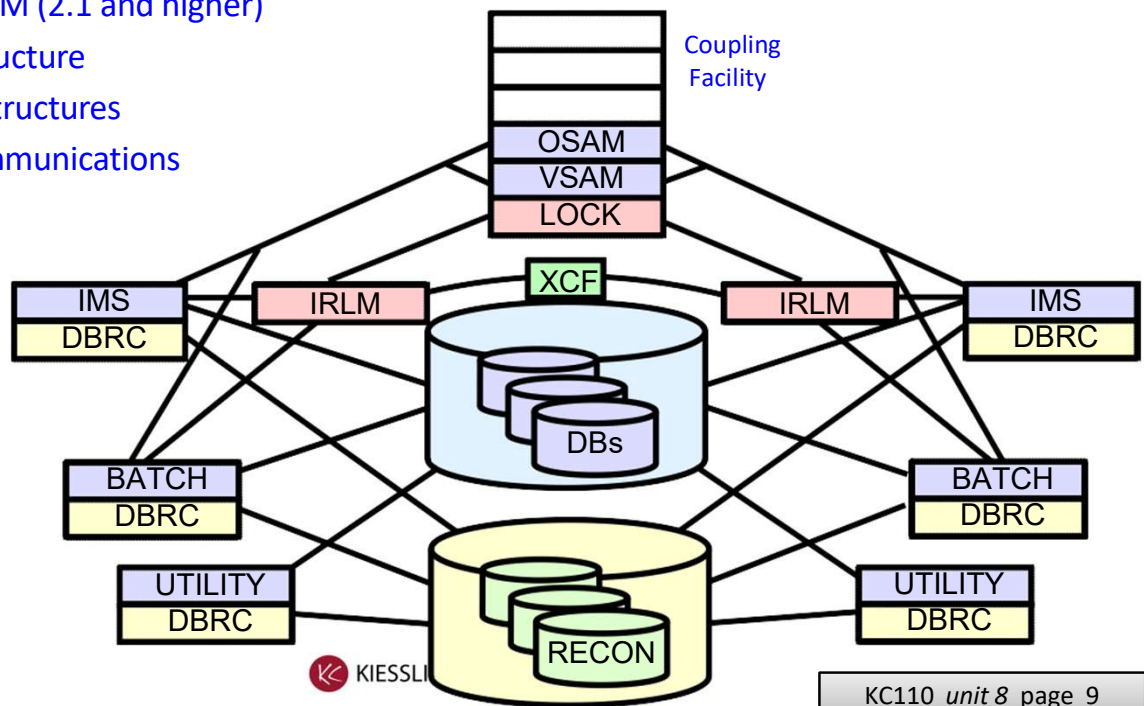
KC110 unit 8 page 8

Notes:

Share Level 3 finally brings additional capacity. With the initial IRLM, there were limitations: Database updates are possible from only 2 processors, and the Pass the Buck process was very inefficient and slow, especially with invalidate buffer processing.

IMS/ESA V5 : *N-way Data Sharing* with improved capacity and performance

- IMS/ESA V5 started to exploit XCF, the Parallel Sysplex, and Coupling Facility structures:
 - New IRLM (2.1 and higher)
 - Lock structure
 - Cache structures
 - XCF communications



KC110 unit 8 page 9

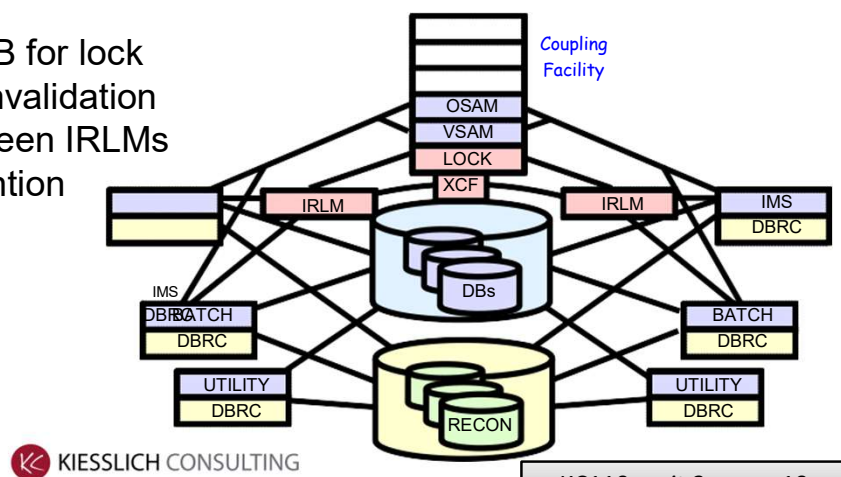
In IMS V5, the technology changed to take advantage of parallel sysplex functionality. This technology was initially delivered as dedicated hardware (processors) that were called Coupling Facilities (CF) and stored data in Structures managed by these CFs. There are different types of CF Structures with several types of organizations that can all be accessed very quickly. Subsequently, Coupling Facilities Code was provided the ability to run on general-purpose IBM S/390/zSeries processors.

A new IRLM (now called the IBM RLM or Inter Region LM) was delivered which uses XCF communications services and XES lock services to manage locks and store them in Coupling Facility Lock Structures. IMS also began to use XES cache services for buffer coherency (buffer invalidation). In IMS V5, these Cache Structures were used only for buffer invalidation, but no data was cached in the CF structures.

From an implementation standpoint, the DBRC definitions remained the same as the implementation that used the old IRLM – databases needed to be registered to DBRC and have a SHARELVL specified if Data Sharing was required. Minor changes were required in the IRLM setup.

IMS/ESA V5: *N*-way Data Sharing with improved capacity and performance (2)

- Issues addressed in IMS/ESA V5:
 - Increased capacity
 - More IMSs could participate in data sharing:
 - Up to 32 IRLMs
 - > Current limit is 64
 - Up to 256 IMSs
 - Improved performance:
 - CF access faster than PTB for lock management and buffer invalidation
 - XCF communication between IRLMs faster than PTB for contention resolution, and so on



KISSLICH CONSULTING

KC110 unit 8 page 10

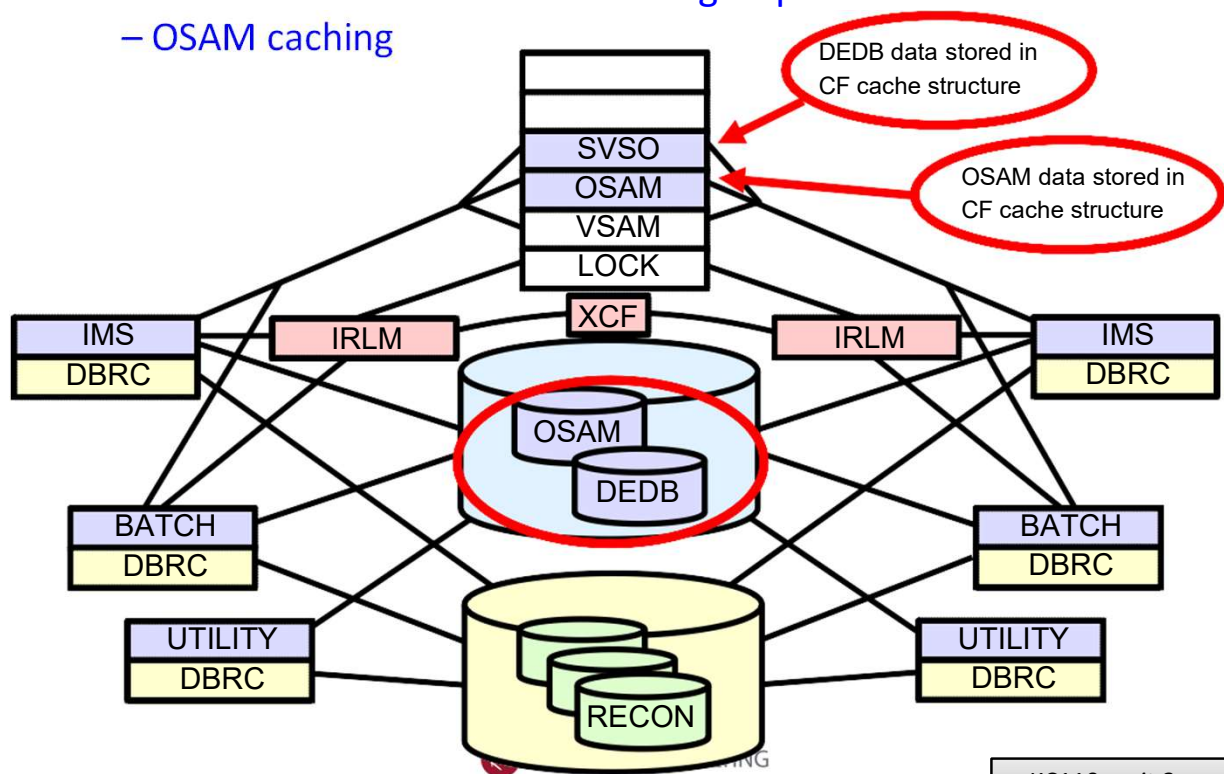
Notes:

The number of IRLMs in a data sharing group increased from 2 to 32 (the limit on the number of connectors to a lock structure). The number of IMS systems sharing data was 256 (the limit on the number of connectors to a cache structure).

The use of XCF communications, together with the improvements that XES lock services and cache services provided, dramatically improved the performance of IMS data sharing.

IMS/ESA V6: DB and TM resource sharing (1)

- IMS/ESA V6 added additional data sharing support:
 - Shared DEDBs with Virtual Storage Option
 - OSAM caching



KC110 unit 8 page 11

Notes:

Version 6 added to IMS's exploitation of parallel sysplex data sharing by supporting the caching of IMS data in cache structures.

OSAM used the store-through structure to keep copies of data in the structure AFTER it was written to DASD. The intent was to minimize the impact of buffer invalidations, but it also allowed the use of the cache structure as a global buffer pool.

Shared DEDB VSO areas used the store-in structure to keep copies of data BEFORE they were written to DASD. At sync point time, committed updates would be written to the structure but not written to DASD until system checkpoint. This greatly improved the performance of DEDB update processing.

IMS/ESA V6: DB and TM resource sharing (2 of 2)

- IMS/ESA V6 added additional transaction manager and operations support:
 - Shared message queues for Full Function and Fast Path EMH:
 - Single set of message queues for all IMSs
 - Queues stored in CF list structures
 - New Common Queue Server (CQS) address space to manage the queue structure
 - CQS uses system logger to log updates to queue structure
 - VTAM Generic Resource support
 - Single-system image to end user (LOGON IMS)
 - Sysplex communications
 - Use of CRC from E-MCS console to send commands to all IMSs in Sysplex and receive responses
 - Automatic Restart Manager support
 - Restart IMS following IMS or MVS failure

Notes:

Also in IMS V6, several new features were introduced.

Shared Queues allowed the sharing of IMS input and output messages across all IMSs in the shared queues group. This allowed one IMS to pick up some of the workload of another busy, or failed, IMS. The end user was never aware of which IMS his transaction executed on. Shared queues used a List Structure in the CF, and a server address space called CQS to access the structure, similar (but not the same) to the way the DLISAS address space is used to access databases.

VTAM Generic Resources was supported which allowed an end-user to log on to a generic name for the IMS data sharing or shared queues group. VTAM would route the logon request to any active IMS.

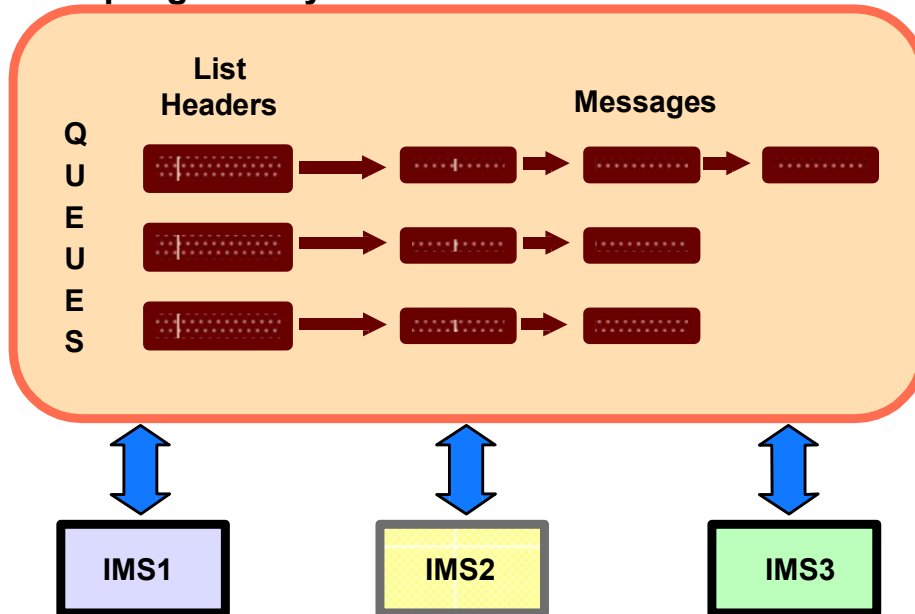
The use of a Command Recognition Character (CRC) was supported to allow the routing of commands to any or all IMSs from an MCS or E/MCS console.

And finally, IMS registered with the Automatic Restart Management function of XCF to enable automatic quick restart of any IMS control region, CQS, or IRLM after abend or system failure.

What are Shared Queues?

A set of input and output message queues which can be shared by multiple IMSs in a Parallel Sysplex.

Coupling Facility



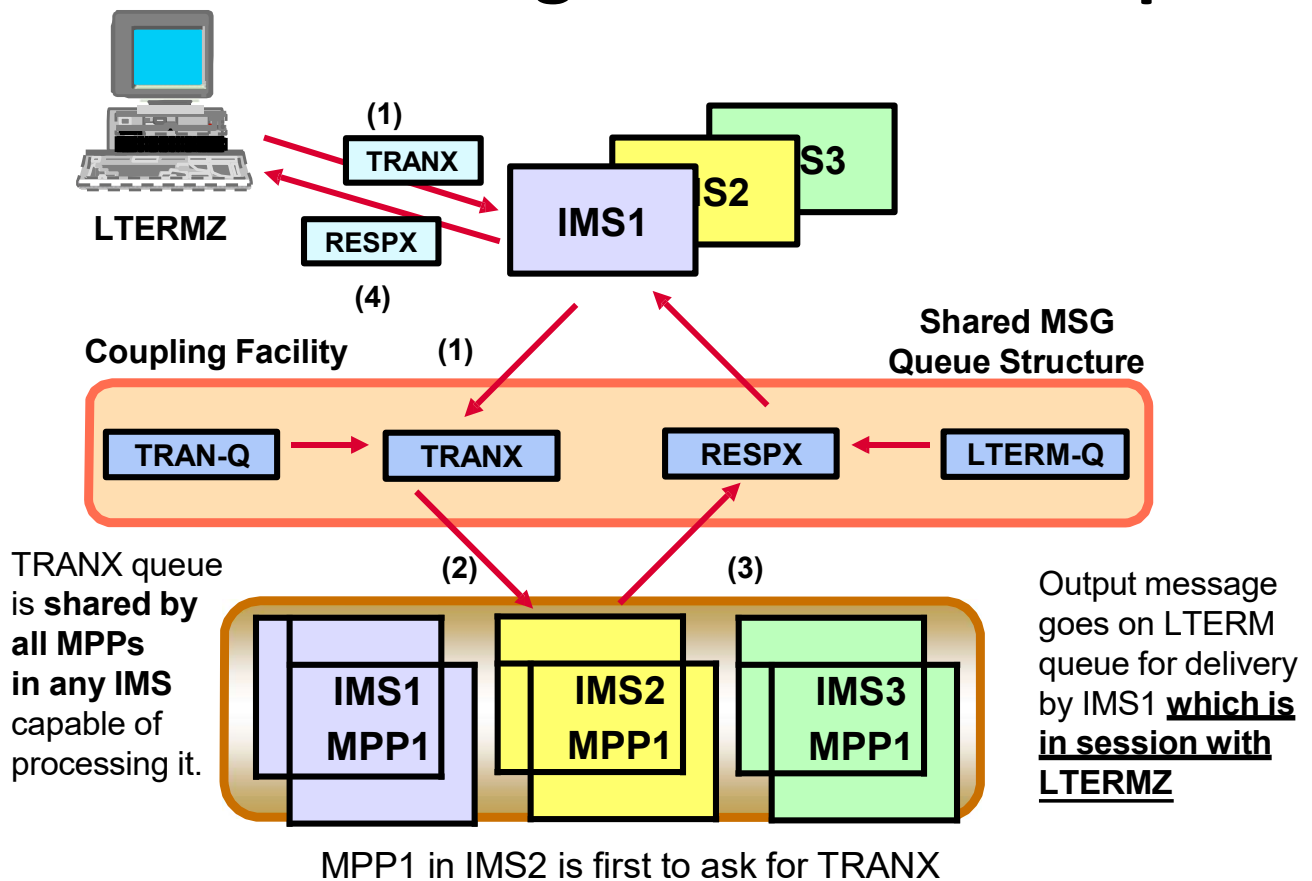
KISSLICH CONSULTING

KC110 unit 8 page 13

Notes:

In this illustration, three IMS systems are able to independently read and write common message queues stored in a List Structure in a Coupling Facility.

Shared Message Queues: Example



Notes:

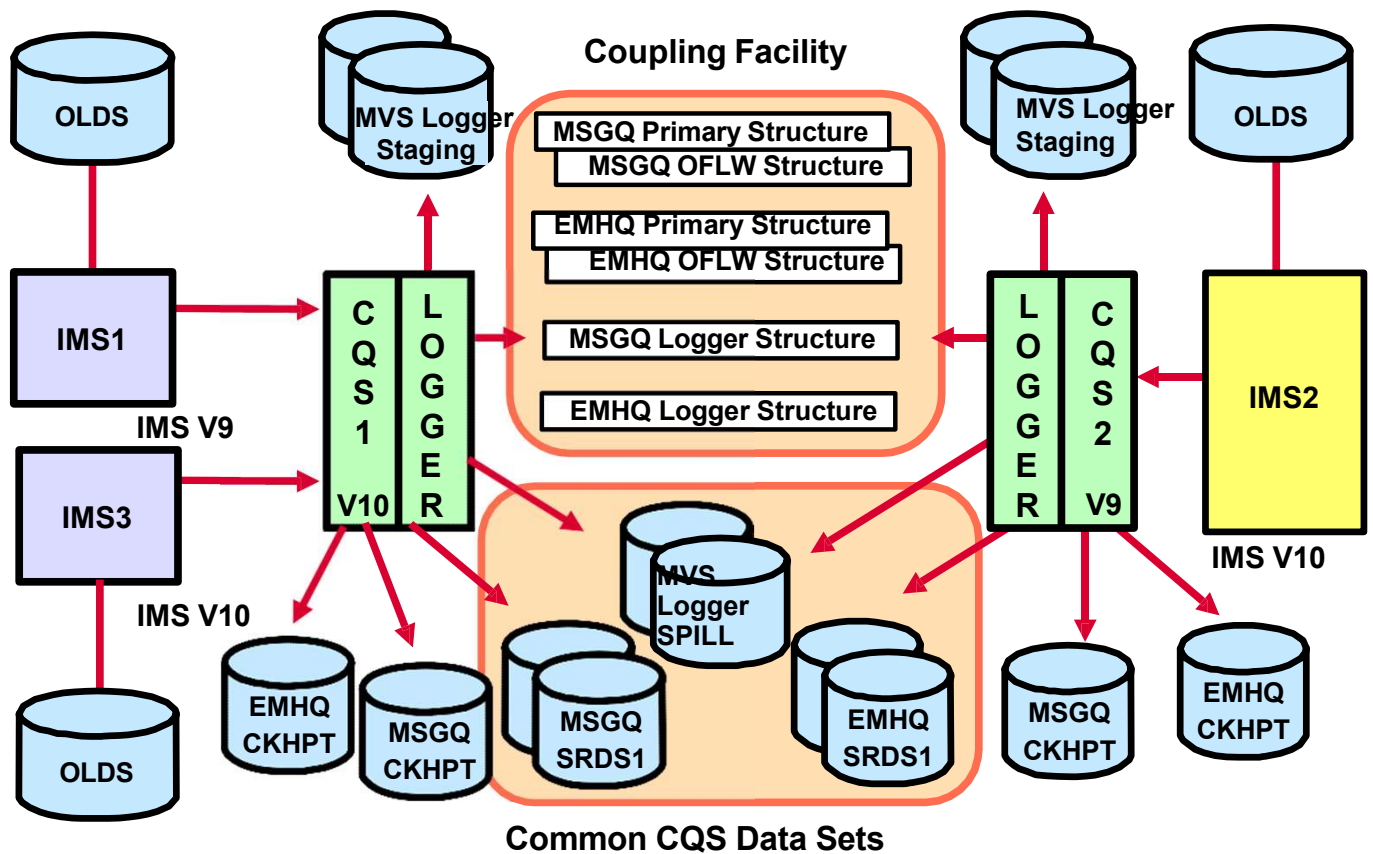
Here we see input messages (TRANX) being received from the device assigned to LTERMZ which is connected to IMS1.

Although IMS1 Queues the message in the Shared Queues, the first MPP that expressed the ability to process the message is MPP1 that runs on IMS2. In some cases, there can be a race condition in which multiple MPPs from different IMS's compete to be the first to retrieve the message.

After IMS2-MPP1 completes processing the message, the response (RESPX) is placed on the Shared Queues by *IMS2*.

At some later time, this response is retrieved from the CF by IMS1 and delivered back to LTERMZ.

Shared Queues Components



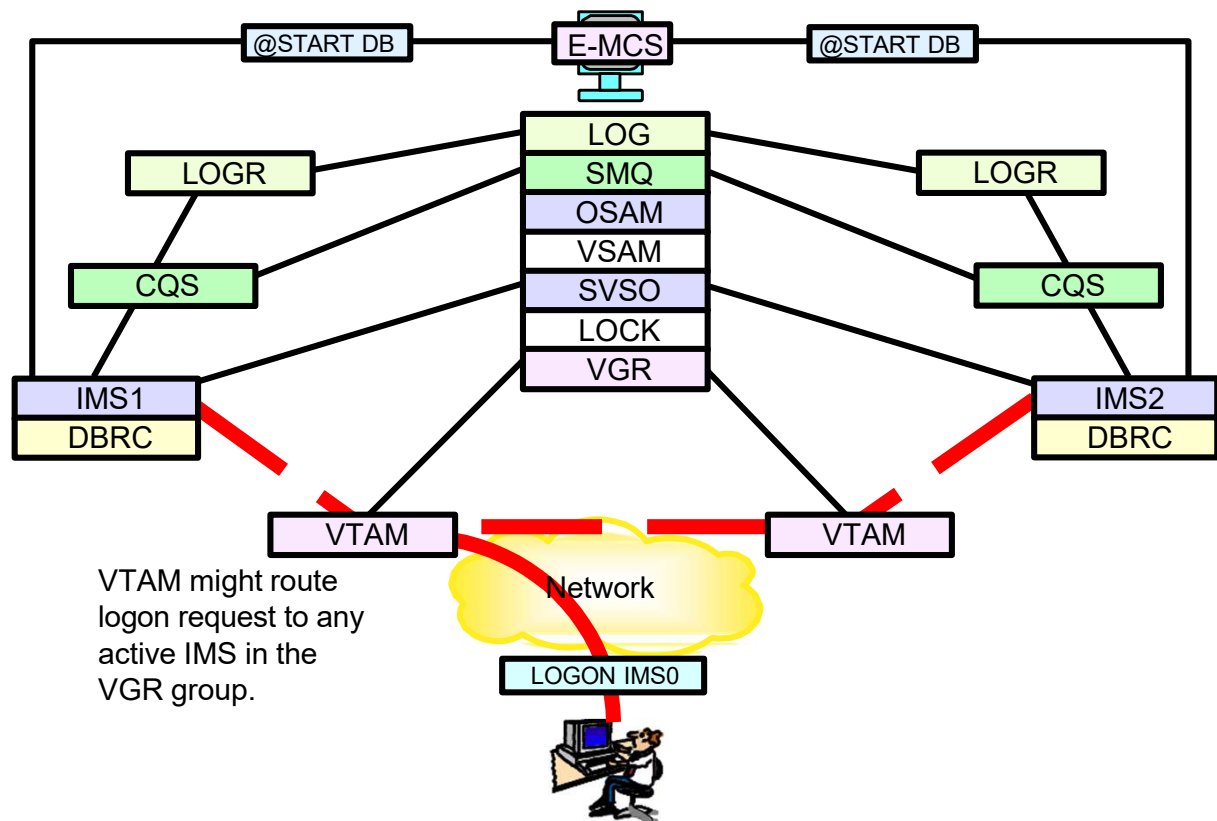
Notes:

Note that IMS does not directly access the Shared Queues Lock Structures in the Coupling Facilities. There is now a new, optional Common Queue Server (CQS) address space that acts as a server for IMS CF access for access to the queues. As a result of this CQS address space, the internal changes to the IMS Queue Manager were relatively modest.

Note that CQS has its own checkpoint data sets and log (logger structures, also in the Coupling Facility) that can be used for recovery/backout of the contents of the Queues in the CF by that CQS system. Also note that in current versions of IMS, multiple IMS systems can connect to a CQS and that the IMS systems, and CQS, do not have to be at the same IMS Version level.

Shared Queues is covered in detail in IMS Shared Queues class, CM61xx / CM621x .

Additional Resource Sharing (Since V6)

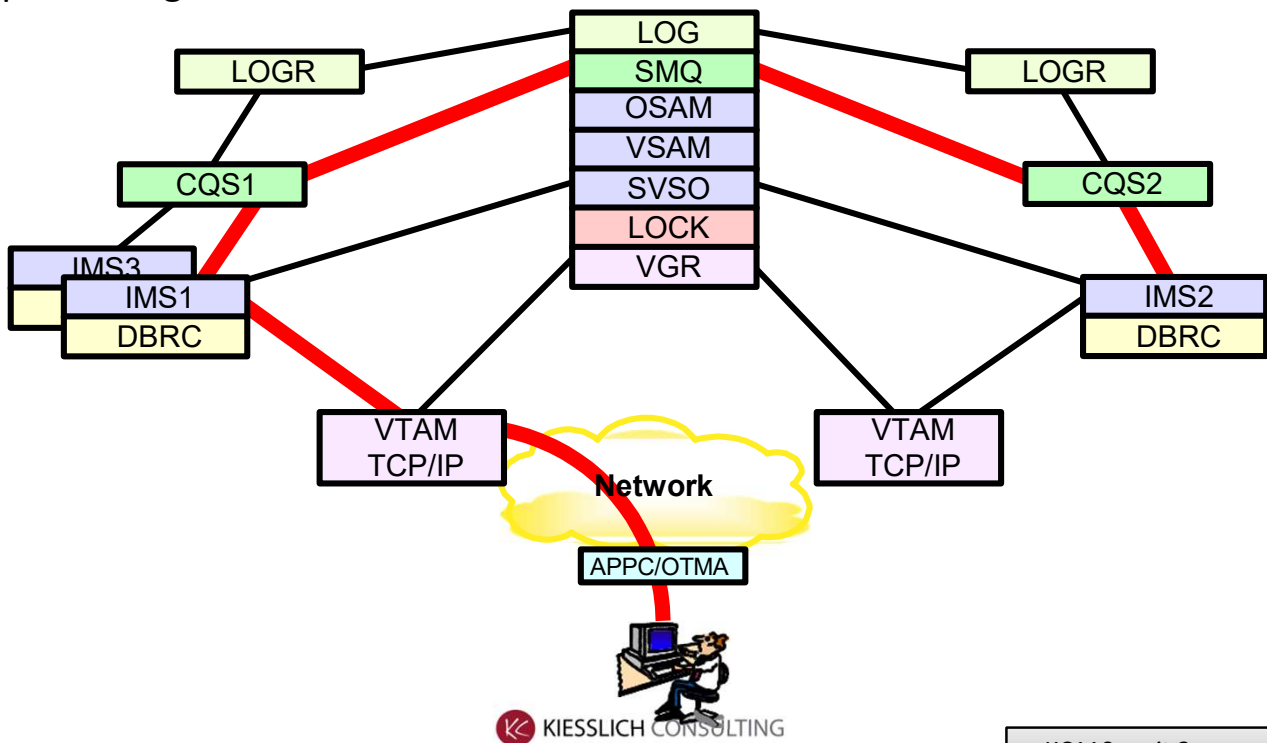


Notes:

This diagram just shows IMS with CQS, the system logger, the CF structures, and VTAM generic resources. An E-MCS console is shown sending a command to all IMSs with a CRC=@.

IMS7+ Added Enhanced SharedQ Support

- Asynchronous APPC and OTMA transactions became eligible for Sysplex-wide processing



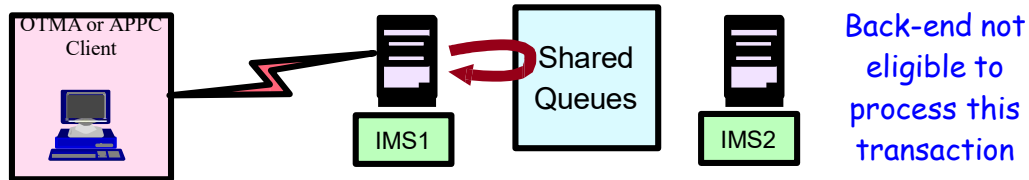
Notes:

Shared queues lacked support for transactions entered from APPC or OTMA sources. Although the transactions were put on the shared queue, they could only be executed on the front-end IMS (the one which received the input). This, of course, eliminated all of the shared queue benefits for these types of terminals.

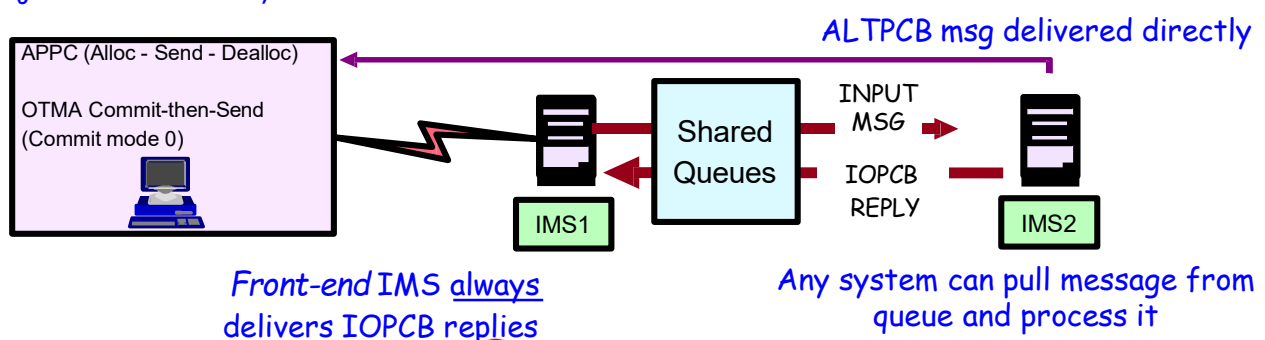
Version 7 added support for asynchronous APPC and OTMA input, allowing these transactions to execute on any capable IMS in the shared queues group.

IMS V7+ Enhanced SharedQueues support for asynch APPC/OTMA msgs

- IMS V6: Introduced Shared Queues support
 - All APPC and OTMA messages processed on SQ front-end



- IMS V7: Enhanced SharedQueues support
 - Asynchronous APPC/OTMA messages could process on any system in the SharedQueues group (front- or back-end)



KIESSLICH CONSULTING

KC110 unit 8 page 18

When Shared Queues support was introduced in IMS V6, the processing of all APPC and OTMA messages 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.

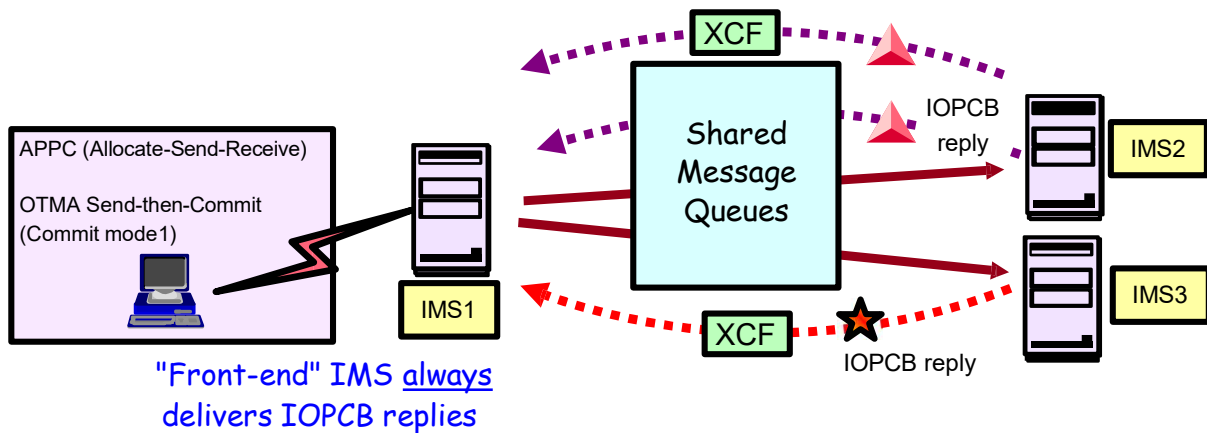
With the introduction of IMS V7, the support was enhanced to allow asynchronous messages to be processed by any IMS in the group. This continues to be the way asynchronous messages are processed in IMS V8. For asynchronous support, IMS queues input messages to the appropriate transaction ready queue 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, therefore, must be able to establish communications with the APPC/OTMA client, meaning that all systems must be APPC and/or OTMA enabled.

IMS V8+ Shared Queues Support for Synchronous APPC/OTMA msgs



- ★ Non-conversational IOPCB reply messages (less than 61K) are sent to the front-end using XCF services.
- ▲ Conversational IOPCB reply messages or any messages greater than 61K are sent to the front-end using Shared Queues along with a special NOTIFY message that is sent using XCF.

With IMS V8, the restriction on synchronous messages has been removed. 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, for example, 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. 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 abort.

Note the following:

All conversational IOPCB reply messages and messages where all segments added

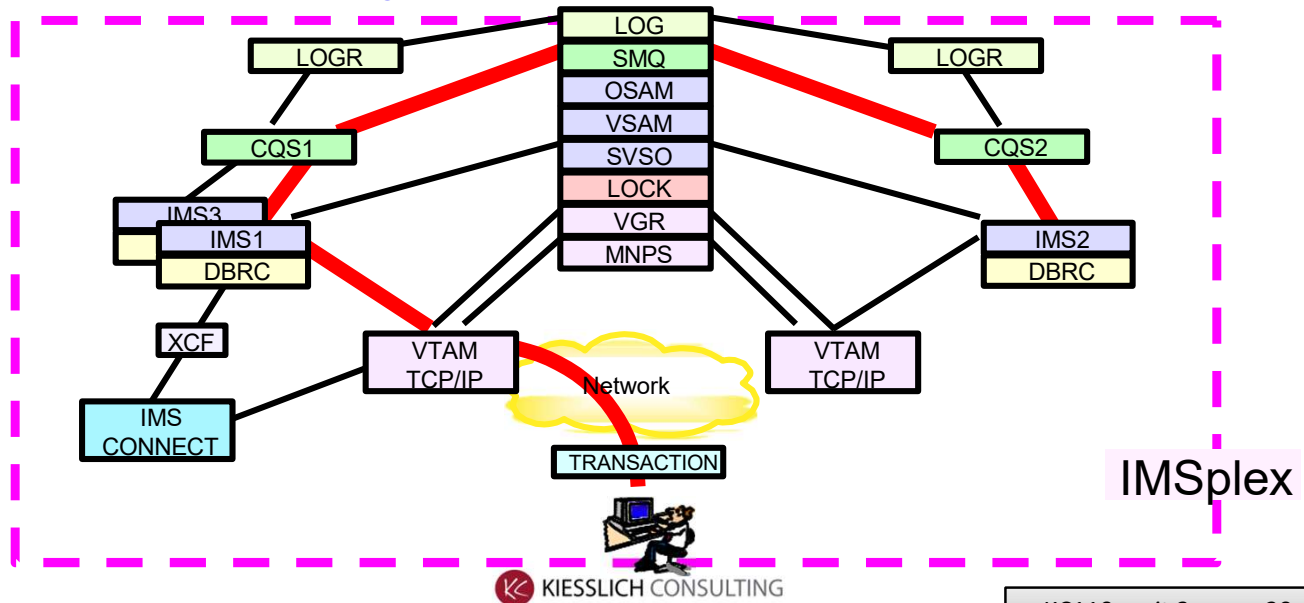
together are greater than 61K, are routed through the Shared Queues with a special notify sent via XCF. A specialized OTMA/APPC task in the front-end reads the notify message which contains the message output queue name, and notifies the appropriate task to retrieve the message from the Shared Queues.

All non-conversational IOPCB reply messages less than 61K are sent to the front-end using XCF services.

If Synchronous Shared Queue Support is used, the z/OS Resource Recovery Services (RRS) is required as the commit manager since the unit of recovery spans IMS systems.

Widespread IMS Resource Sharing evolves towards *The IMSplex*

- Additional features in IMSV7 did allow to exploit many parallel sysplex functions to share resources as an **IMSPLEX**
 - Data sharing, shared queues
 - VTAM generic resources, multinode persistent sessions
 - Automatic restart management, XCF communications



By the time of the introduction of Version 7, IMS had exploited the parallel sysplex for a multiple data sharing and connectivity functions. These include:

Data sharing

Shared queues, which also uses the MVS System Logger

VTAM generic resources

VTAM multinode persistent sessions (IMS's implementation is called Rapid Network Reconnect)

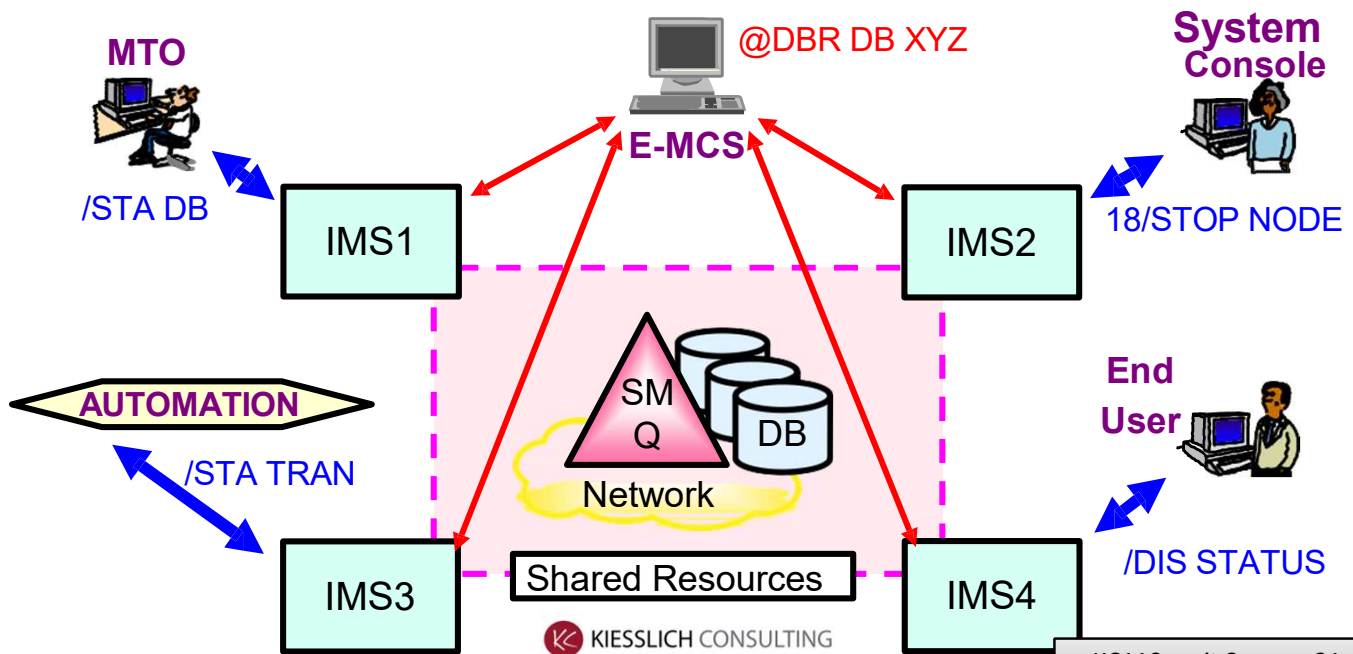
XCF communications (IMS Connect and intra-IMSplex communications)

Plus a variety of XCF services (group services, signalling services, monitoring services).

The addition of this additional capability also added complexity.

Managing Shared IMS Resources

- Although IMS systems increasingly shared resources and processing, managing these resources became more difficult:
 - The sharing IMS systems persisted in being individually controlled
 - Systems management functions needed to be more robust



KC110 unit 8 page 21

Notes:

The reason for all this additional capability, and complexity, is to share resources across multiple IMSs in a parallel sysplex, or IMSplex.

But while this has improved availability, capacity, and performance, it has at the same time made the task of managing these resources more difficult. For example, IMS commands used to control the resources must be entered to each IMS individually and, with a few exceptions for databases, take effect only on the IMS where the command is entered. IMS V6 began support of the Command Recognition Character so that a command entered from an MSC or E-MSC console can be routed to all IMSs, but this is really not part of IMS, and is not particularly user-friendly.

As the number of IMSs in the IMSplex grows larger, management of these resources becomes more difficult.

Better Systems Management was clearly needed

- Better resource management:
 - Address the management of terminals, transactions and users throughout an IMSplex. Inconsistent definitions should also be prevented:
 - Sysplex terminal management
 - Coordinate the online change process across all IMSplex members
 - Global process management
 - Give exits the ability to determine terminal/user status globally
 - Global callable services
- Better operations management
 - Facilitate operational control of IMSplex members:
 - Single Point of Control
 - Global automation

Better capabilities for managing the operations of the IMSplex and its resources were needed. Version 8 identifies two areas for improvement:

Resource Management

Sysplex terminal management applies to the management of terminals, transactions and users within the IMSplex. In the absence of this, a resource with name PAYROLL could be defined as an LTERM in one IMS and as a transaction in another. Global process management is the coordination of processes across all members of the IMSplex. An example is Global Online Change. Global callable services allows user exits to determine the status of terminal, LTERM and user resources IMSplex-wide.

Operations Management

Single point of control is one or more entry points into the IMSplex where commands can be entered to any or all IMSs and consolidated responses from those IMSs can be received.

Global automation is an interface which allows user- or vendor-written automation software to issue commands and receive consolidated responses.

The IMSPLEX

- Definition of an IMSplex:
 - An IMSplex is a set of IMS address spaces that are **working together as a unit** and are most likely running in a parallel sysplex hopefully with a **Common Service Layer (CSL)** to assist in its management
 - **Note:** The concept of the IMSplex was not new; however the term was first formalized at the time of IMS V8.
 - Examples of an **IMSplex** include ...
 - A set of IMS control regions at the V8 and/or V9 and/or V10 level **without** a CSL that are data sharing or message queue sharing
 - This is arguably the worst case; we still have all the management complexity but without a CSL to assist us
 - A set of IMS control regions at the V8 and/or V9 and/or V10 level **with** a CSL that are data sharing or message queue sharing
 - A single IMS control region at the V8, V9 or V10 level with a CSL
 - This configuration makes sense especially for an IMS V10 (or higher) system since a CSL is required for the Dynamic Resource Definition (DRD) function

Notes:

The IMSplex is not a new concept, nor even a new term. But it has been formalized in IMS V8. It is what you have always thought it was - a group of address spaces (related to IMS, of course) that work together to perform a set of services for end users. Although the term was introduced formally in IMS V8, the concept pre-dated IMS V8. Currently, a IMSplex can consist of a mixture of IMS V8, IMS V9, and IMS V10 IMS systems. And although we might think of the IMSplex running only on a parallel sysplex, it is technically not a requirement.

Common Service Layer

- New IMS **address spaces** built on Base Primitive Environment:
 - **Structured Call Interface (SCI)**
 - IMSplex member registration
 - Communications between IMSplex members
 - **Operations Manager (OM)**
 - IMSplex-wide command entry and response
 - **Resource Manager (RM)**
 - Global resource and process management
 - Optional CSL Address Space since IMS V9
 - VTAM terminal/user status recovery
- Enables new systems management functions in IMSplex:
 - **Sysplex Terminal Management (STM)**
 - Uses SCI and RM
 - **Single point of control (SPOC) and user-provided automation (AOP)**
 - Uses SCI and OM
 - **Coordinated Online Change (Global Online Change)**
 - Uses SCI, OM, and RM
 - **Dynamic Resource Definition (DRD)**
 - Uses SCI, and OM



KISSLICH CONSULTING

KC110 unit 8 page 24

The previously existing IMS architecture did not provide a good means of providing Resource or Operations Management functions across multiple IMS Systems. So in IMS V8, the IMS architecture took a major evolutionary step. The architecture enhancement introduced by IMS V8 is called the Common Service Layer, or CSL, and consists of three new address spaces:

Structured Call Interface Address Space

SCI allows for members of the IMSplex to register as members. Members include not only the IMS control regions, but might also include CQS, DBRC, SPOC and Automation programs and, of course, the new CSL address spaces.

SCI provides for communications between members. Registered members can use SCI services to communicate with other members of the IMSplex.

Operations Manager Address Space

OM provides the API for a SPOC or Automation Program to gain access to the IMSplex, enter commands, and receive responses.

Resource Manager Address Space

RM provides the infrastructure for global resource and process management – In IMS V9 (and later) this Added Address Space is optional unless specifically used by one of the sysplex management functions.

RM also supports terminal and user status recovery between IMSs.

These address spaces provide the infrastructure which enables members of an IMSplex with CSL to implement systems management functions including:

Sysplex terminal management (STM)

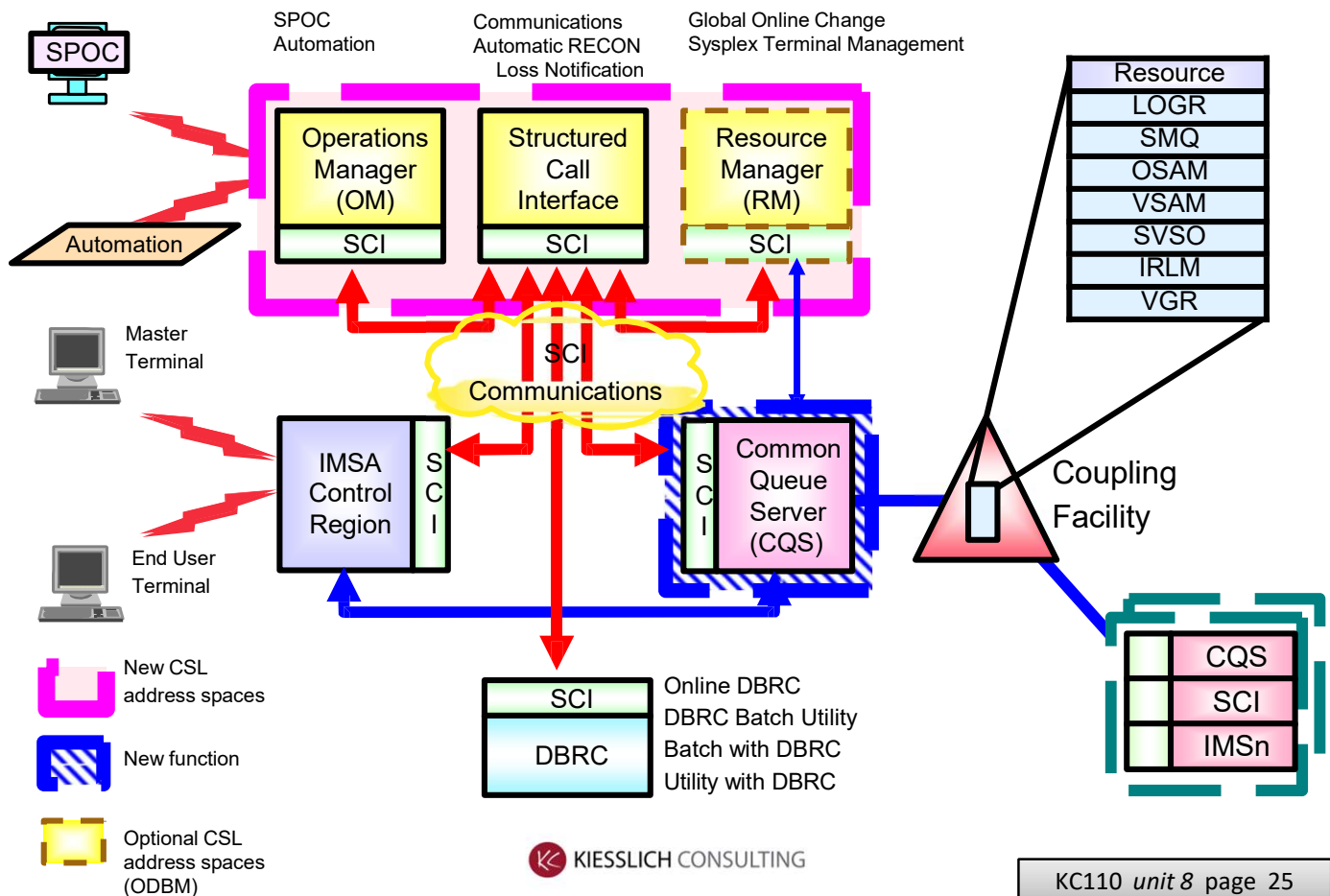
Single point of control (SPOC)

Coordinated online change, also called global online change (G-OLC)

Dynamic Resource Definition

Each of these functions uses one or more of the CSL address space services.

CSL Architecture



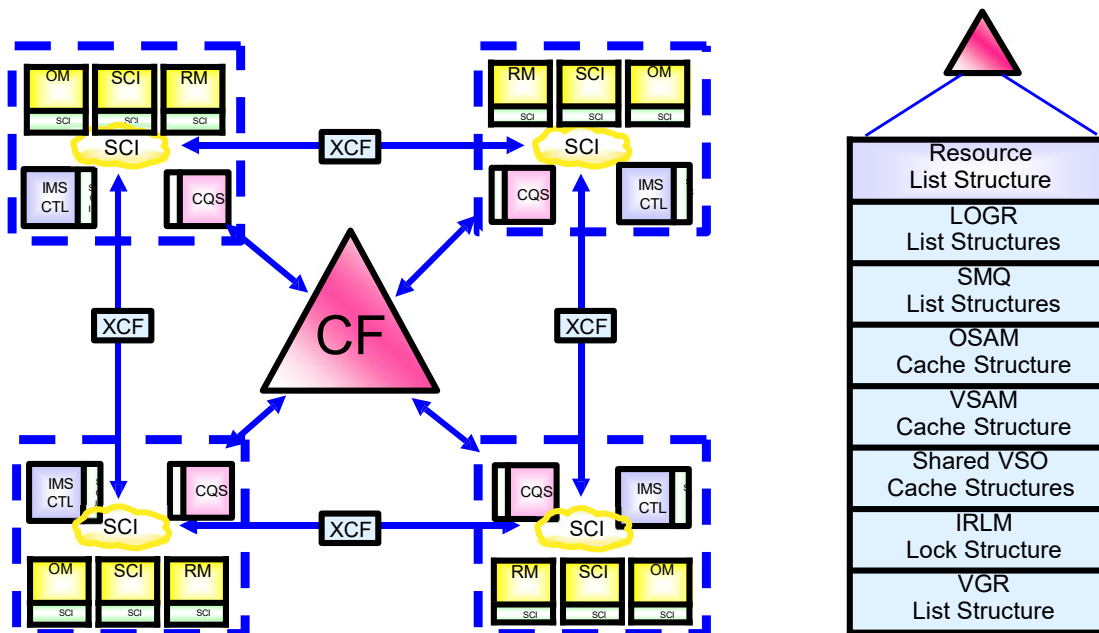
Notes:

This diagram shows a minimum configuration for IMS in a CSL environment utilizing all functions. The three new address spaces are shown with the function they support written above them. For example, OM supports SPOC and Automation.

Sysplex terminal management requires a new structure in the coupling facility called the Resource Structure. CQS, the Common Queue Server, is used by the RM address space to access this structure. The same CQS can also be used to access the Shared Queues structures. If STM is not desired, there is no need for a Resource Structure. If a Resource Structure is used, then all RMs in the IMSplex must use the same one. The interface between IMS and CQS for SQ and between RM and CQS for STM continues to use the CQS interface introduced with IMS V6 - it does not use SCI. All other communications uses SCI.

All forms of DBRC can register with SCI to get Automatic RECON Loss Notification (ARLN). This is a function which allows all DBRCs to be notified when a RECON is lost and the SPARE is used. More on this later.

IMSplex Configuration



- In an IMSplex:
 - All members share the same CF structures
 - Intra-IMSplex communications is implemented by SCI using XCF across OS images

When there are multiple IMSs within the IMSplex, each IMS (actually each z/OS image) must have a copy of the SCI address space.

Only one copy of OM (and RM if used) is required per IMSplex, although for availability and performance, it is recommended that one per z/OS image be started. Wherever there is an IMS with shared queues, or an RM with a Resource Structure, there needs to be a CQS per z/OS LPAR. A single CQS can serve for access to both the Shared Queue structures and the Resource structure for the IMS systems executing on a z/OS system.

When it is necessary to communicate between members of the IMSplex (for example, between IMS and RM), SCI is used. If the address spaces are not on the same z/OS image, XCF communications services are used by SCI; within a z/OS image, SCI uses cross memory services.

The IMS V10 IMSplex Administration Guide (SC18-9709-00) should be extensively reviewed before attempting to establish an IMSplex.

The CSL Operations Manager component

- Provides an API supporting common point of command entry
 - Focal point for operations management and automation
 - Command responses from multiple IMSs are consolidated
- Provides the following services to members and clients of an IMSplex
 - An API for IMS Commands submitted from outside IMS
 - Type 1 (Classic) IMS commands (/cmd ...)
 - Type 2 (New) IMSplex commands (QRY, INIT, TERM, DEL, UPD)
 - Command Registration to support any command processing client
 - Clients tell OM which commands it can process
 - Command Security
 - Perform authorization within OM - before sending to IMS
 - RACF or user-written command security exit
 - Command Routing to IMSplex members registered for the command
 - Command Response Consolidation from multiple individual IMSplex members into a single response to present to the command originator

The Operations Manager (OM) is a server address space in the IMSplex. It acts as a common point of command entry into the IMSplex. That is, a command can be entered through the operations manager interface to any IMS in the IMSplex. The responses to those commands will be consolidated by OM and returned to the client (service requestor). OM, with its clients, can become the focal point for operations management and automation within the IMSplex [note that there might be multiple OM clients].

The major services provided by OM are listed:

OM provides a documented API for IMS commands to be entered by an OM client to any or all IMSs in the IMSplex and to retrieve consolidated responses. The API is documented in the CSL Command and Reference manual. Any type of IMS command can be entered. This includes the classic commands (those starting with a '/') as well as the five new IMSplex commands (we'll discuss these later).

OM can support any command process client that conforms to the API - not just IMS. In an IMSplex, the Resource Manager (RM) is also a command processing client, although there is only one command it can process.

OM can provide authorization processing for IMS commands before that command is submitted to IMS. Authorization can be done using RACF, a user exit, or both. Or the

user can opt not to perform command authorization in OM. There is a new parameter in IMS which indicates whether IMS itself should perform authorization processing on commands entered through OM. Presumably, if OM does command authorization, it would not be necessary for IMS to also do it.

An OM client can instruct OM to route a command to any or all registered IMSs. IMS, during initialization, will inform OM what commands it wants to process by registering that command with OM. If a client enters an invalid command (for example, /DBX), then OM would reject it because it was not registered. It would not be passed on to IMS.

And finally, OM will consolidate command responses before returning them to the client.

Type2 IMSPLEX commands

- INIT (INITiate process)
 - INIT OLC: Starts a global online change (G-OLC) process
- TERM (TERMinate process)
 - TERM OLC: Stops a global online change that is in progress
- CRE (CREate resource): Command added in V10 for DRD
 - CRE DB NAME(name) <other attributes for DRD defined DB>
 - CRE PGM NAME(name) <other attributes for DRD defined APPL>
 - CRE RTC NAME(name) <other attributes for DRD defined FP RCTE>
 - CRE TRAN NAME(name) <other attributes for DRD defined TRAN>
- UPD (UPDate resource):
 - UPD LE: Updates dynamic LE runtime options
 - UPD TRAN: Updates selected TRAN attributes
 - UPD <DB | PGM | RTC > Added and *UPD TRAN* enhanced for DRD
- DEL (DElete resource):
 - DEL LE: Deletes dynamic runtime LE options
 - DEL <DB | PGM | RTC | TRAN> Added for DRD

This page shows five of the six current (as of V10) IMSplex commands; more are planned. These commands can only be entered through the OM interface.

INITiate: This command is used to initiate an IMSplex global process. In V8, the only global process in Global Online Change.

TERMinate: This command is used to terminate a global process when something goes wrong. For global online change, this would be the equivalent of the /MODIFY ABORT command.

CREate: was added in V10 to support DRD definition of resources. Although not shown here, there are also descriptor DRD entries (for example, DBDESC, PGMDESC, and so on) that can be used as templates for defining various resources through DRD

UPDate: This command is used to update a resource. In V8, two resources can be updated - the dynamic LE runtime parameters, and some TRANSACTION attributes. In IMS V10, this command was enhanced to provide DRD support to change attributes previously set.

DElete: This command is used to delete a resource. In V8, the only resource that can be deleted is the dynamic LE runtime parameters added earlier with the UPD LE command. In IMS V10, this command was enhanced to provide DRD support to delete DRD defined resources

When IMS processes the UPDATE TRAN command it writes a new X'22' log record. IMS

does not write a X'02' log record because the condensed command buffer is not used.

DFSLOG22 DSECT maps the X'22' log record.

Type2 IMSPLEX QueRY command

- QRY IMSPLEX: Returns information about one or more members of the IMSplex
- QRY MEMBER: Returns status and attributes of the IMS members in the IMSplex
- QRY LE: Returns runtime LE options
- QRY OLC: Returns OLC library and resource information
- QRY TRAN: Returns TRAN info similar to /DIS TRAN
- QRY DB|PGM|RTC|TRAN command added or enhanced to provide information similar to /DISplay command
- QRY STRUCTURE: Returns structure information for the RM resource structure

The QueRY command has several parameters. All of the QRY commands include a SHOW parameter to specify what fields you want returned. Or you can say SHOW(ALL).

IMSPLEX displays information about the IMSplex itself. It includes for each member of the IMSplex: its MVS name, jobname, status, type (for example, IMS, DBRC, OM, RM, AOP, and so on) and subtype (for example, DBDC, DCCTL, DBCTL, FDBR).

MEMBER displays information about a specific member by member type. For V8, TYPE(IMS) is the only type supported. It shows the status some of the attributes of each member of that type, such as the global online change status for each IMS, or whether this IMS is using Shared Queues.

LE displays the dynamic LE runtime parameters set by an earlier UPD LE command, including any filters and what the LERUNOPTS are.

OLC displays information from the new OLCSTAT data set used for global online change. This information includes the OLCSTAT data set name, the suffixes for the active OLC libraries, the OLC modid, the type of the last OLC, and all IMS members who are currently in the global online change group and current with the active libraries.

TRAN displays transaction attributes. The information displayed is equivalent to the /DIS TRAN command output, except that it can show the output from multiple IMSs.

It also shows the global queue count if those IMSs are in a shared queues group.

The command also supports QRY for DB, PGM, RCT and reports information similar to the corresponding Type 1 /DISPLAY command.

STRUCTURE displays information about the Resource Structure, including the number of entries and elements allocated and in use, and the Entry-to-Element ratio.

UPD / QRY TRAN example

```
UPD TRAN NAME(PART) SCOPE(ALL) STOP(Q,SCHD)
START(TRACE) SET(CLASS(4))
```

TRANCODE	MBRNAME	CC
PART	IMS1	0
PART	IMS2	0
PART	IMS3	0

Actual response is in XML format.
Formatting for display is the
responsibility of the command
originator.

```
QRY TRAN NAME(PART) SHOW(CLASS,STATUS)
```

TRANCODE	MBRNAME	CC	CLS	STATUS
PART	IMS1	0	4	STOQ, STOSCHD, TRA
PART	IMS2	...		

Notes:

This shows an example of an UPD TRAN command. SCOPE(ALL) tells OM to send this command to all active IMSs in the IMSplex. The other parameters are obvious. The response to the UPD command shows that each of the IMSs executed that command. We will cover other UPD commands in the DRD topic.

IMS Type1 *Classic* commands and OM

- Most classic IMS commands (/cmd ...) can be entered through OM API
 - IMS commands specific to an input LTERM or NODE are not supported from OM
 - For example
/SIGN ON|OFF, /EXIT, /REL, /RCL, ...
- If resource structure exists, some commands have global impact, for example:
/STOP NODE ABC
 - Node ABC is flagged as stopped in resource structure
 - Node ABC cannot log on to any IMS in IMSplex

Classic IMS commands can also be entered through the OM interface. Each IMS will register with OM those commands which it will accept. In general, all IMS classic commands can be entered through the OM interface except those that apply to the inputting LTERM. For example, it would not make sense to enter the /SIGN ON command from the SPOC since this is not an IMS terminal. Other examples are shown. Some IMS commands have global impact if Sysplex Terminal Management (discussed later) is active. For example, the /STOP NODE ABC command can stop the node resource in the resource structure. When this is done, that node is stopped on all IMSs in the IMSplex.

IMS Type1 *Classic* commands and OM (2)

- Some commands execute in every IMS where command sent
 - Not aware of IMSplex
 - /DIS TRAN TRX1 QCNT
 - Will execute in each IMS where command is routed
 - All will return same value (global queue count)
- Most commands depend on several factors
 - Command source, RM active with structure, affects significant status, resource exists on structure, resource owned by this IMS, resource owned by another IMS, display or update, ...
- Command differences documented in *Command Reference* manual in the *Usage Notes* section for commands:
 - As noted earlier, *The IMS V10 IMSplex Administration Guide* (SC18-9709-00) should be also be used when establishing or supporting an IMSplex
 - Both these manuals are worth studying!!

Some classic commands execute on every IMS which receives it. These are commands for resource status recovery is not supported by STM. For example, /DIS TRAN will execute on every IMS. Since part of the command output is the global queue count (in a shared queues environment), each IMS will query CQS, which queries the SQ structure, for this count.

Those commands displaying or changing the status of an STM-support resource (for example, node, LTERM, user), execution of the command depends on several factors. For example, if a NODE is logged on to IMS1, the /STOP NODE command can only be executed by IMS1.

For display commands, only one IMS will return global information. For example, for /DIS LTERM XYZ QCNT, only one IMS will display the global queue count. Each IMS will display other local LTERM attributes.

For these commands, OM selects one of the IMSs as the MASTER for that command. Only the MASTER will perform global operations.

There are a significant number of changes to the way IMS commands work in this environment. They are all documented in the Command Reference manual, and should be studied when migrating to a CSL environment.

IMS SPOC - Single Point Of Control an OM client

- Runs under MVS and TSO
 - ISPF Application (DFSSPOC)
- Might or might not be on the same z/OS image as OM:
 - Must be on same MVS as an SCI in the IMSplex
 - Uses SCI to communicate with OM
 - As noted earlier, there are performance and availability benefits in having an OM on all Z/OS systems that host IMSplex participants
- Although called *Single* Point Of Control, a more descriptive word would be *centralized*:
 - SPOC provides a terminal from which IMS commands can be entered by a person to one or more members of an IMSplex
 - There can be multiple SPOC concurrent sessions in an IMSplex
- Formats command responses to be read by a person
 - OM response is encapsulated in XML
- TSO SPOC uses OM for security
 - OM provides security checking interface to RACF
 - TSO userid is used to determine RACF authorization

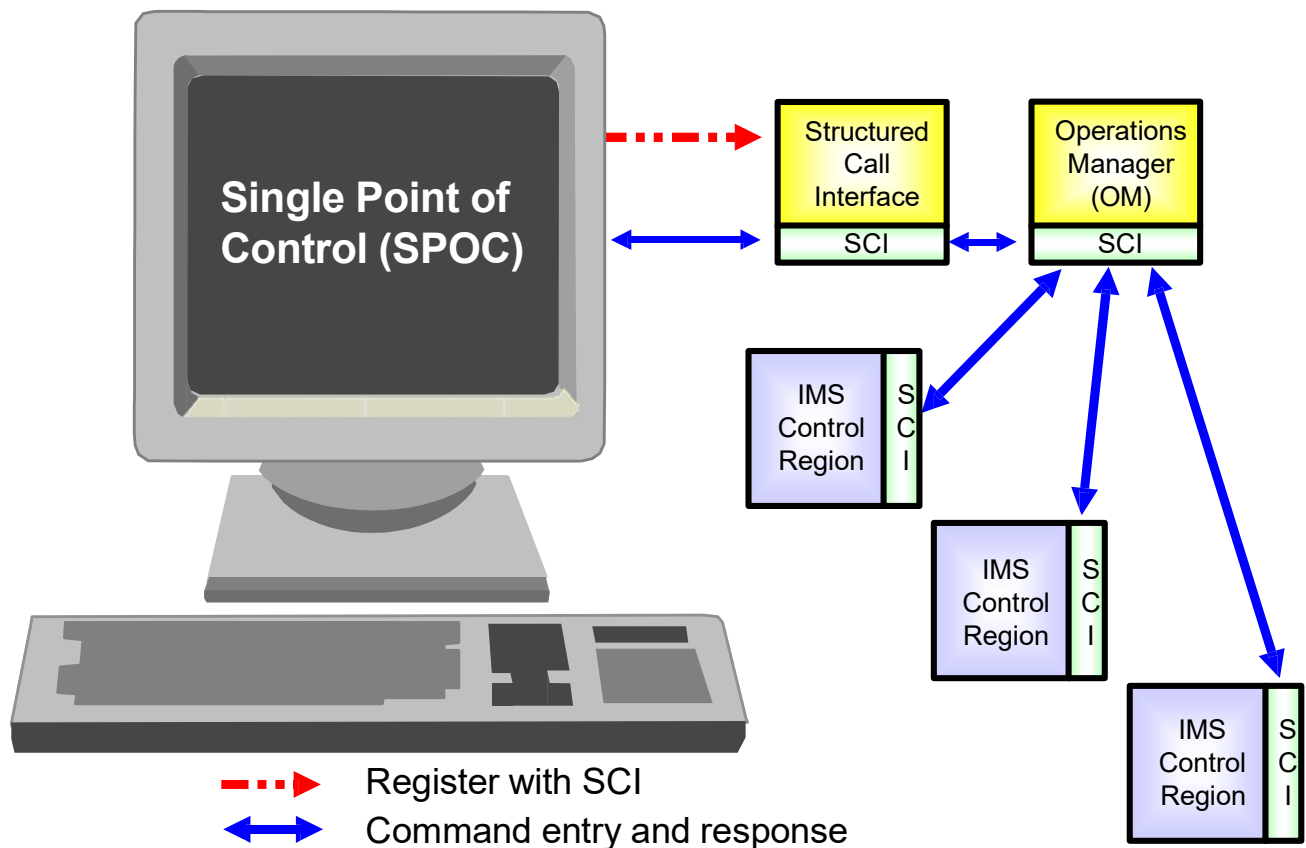
KC110 unit 8 page 33

The IMS TSO Single Point Of Control (SPOC) is an ISPF application that communicates with OM. It is furnished with all IMS versions since IMS V8 and is invoked using the command DFSSPOC. Like other ISPF applications, a split screen can be used to start multiple DFSSPOC applications. The TSO SPOC must be on the same z/OS as an SCI, but does not require that an OM also be present on the LPAR.

Although it is called the Single Point of Control, this does not imply that there can be only a single SPOC. Nor does it imply that a single SPOC can completely control an IMSplex. There are many cases where commands must be entered into IMS as a result of an operator or automation program seeing a DFS message, or as the result of a scheduled time or time interval. A SPOC session only has access to responses from its command input.

The DFSSPOC application enters commands to IMS through OM using the CSLOMxxx interface and receives responses in XML format. It then formats these responses and displays them on the screen similarly to the way IMS would display them. The TSO SPOC does not security checking on its own. This is done by OM and/or IMS using the TSO USERID for authorization.

SPOC registers with local SCI



Notes:

Every SPOC must register with SCI. The SPOC does not have to be on the same z/OS image as OM or IMS, but it does have to be on the same MVS image as SCI. After registering, the SPOC will use SCI to communicate with OM, send commands to IMS, and receive responses for those commands.

IMS SPOC features

- The SPOC allows tailoring by each user to set individual preferences:
 - Users are allowed to specify shortcuts and set default command parameters
 - Groupings of IMSplex members can be specified
 - Users can come up with their own abbreviations for long, and complex, commands
- The SPOC can provide flexibility in the form and the routing of commands:
 - Classic and IMSplex commands can be processed in a SPOC session
 - It also allow the user to enter commands to one or more IMSs
- Command output is enhanced compared to output provided to the tradition MTO or System console:
 - Display consolidated IMSplex and classic IMS command responses
 - A history of commands is maintained
 - Allows the user to restrict IMSplex command output and to sort the IMSplex command response by column

The DFSSPOC application has many nice features to make it easy to use: Setting preferences for command entry; such as the IMS or IMSs to route the command to, whether or not to allow shortcuts and how to process them, a default OM wait time, and so on. An example of the preference screen will be shown in a few pages. Shortcuts - abbreviated way ; a command which is entered frequently can be abbreviated to just a few characters.

IMSs can be defined as part of a group. For example, in an 8-way IMSplex, 4 could belong to one group and 4 to another. Command can then be routed to groups instead of individual IMSs. The user can override the default routing on any command input.

DFSSPOC collects the consolidated responses from OM and displays them, identifying which IMS each response is from. DFSSPOC will keep a short history of recently entered commands and the responses (only within that TSO session), allowing the user to go back and browse early command responses and even to edit and reenter the same command.

Type 2 (IMSplex) commands permit the tailoring of which output columns are returned; with SPOC the responses can be sorted on column value.

IMS SPOC Command Entry panel

File Display View Options Help

IMS Single Point of Control

Command ===> _____

----- Plex . ____ Route . ____ Wait . ____

Response for: QRY IMSPLEX

IMSpplx	MbrName	CC	Member
PLX0	OM1	0	IMS5
PLX0	OM1	0	IMS4

F13=Help F15=Exit F16=Showlog

command line

command response

Notes:

The basic operation is to enter a command in the command line and for the command response to be provided in the data area below the command line.

The command line is cleared in preparation for the next command: the command issued is shown just above the command response. This screen is how the panel would appear after the command was processed.

Command Entry from IMS SPOC

File Display View Options Help

PLX0 IMS Single Point of Control

Command ==> QRY TRAN NAME(A*) SHOW(ALL)

----- Plex . _____ Route . IMS13 _____ Wait . _____

Response for:

Override Preferences



F13=Help F15=Exit F16=Showlog F18=Expand F21=Retrieve F24=Cancel



KIESSLICH CONSULTING

KC110 unit 8 page 37

In this example, we are showing a Type 2 (Query) command being entered. Alternately, we could have entered some Type 1 (for example, /START TRAN...) instead. With either type of command input, the fields described below apply. There are three short fields: plex, route, and wait. These are temporary overrides of the fields in the preferences panel. These values are discarded after you exit SPOC. IMSplex command and response pages with a lot of data will contain a scrolling indicator:

"More: <-+> ".

- < More data to the left.
- More data below.
- + More data above.
- > More data to the right.

Type 2 Command Response example

```
File Action Manage resource: SPOC View Options Help
=====
PLEXi                               IMS Single Point of Cont
Command ===

Response fo: QRY TRAN NAME (A*) SHOW(ALL) More: >
Trancode MbrName CC LPSBname LCls LQCnt LLCT LPLCT LPLCTTime LQ
ADDINV IMSA 0 DFSSA 1 0 2 6553 6553500 7 7
ADDPART IMSA 0 DFSSA 1 0 2 6553 6553500 7 7
AUTRAN11 IMSA 0 AUTPS 1 0 2 6553 6553500 7 7
AUTRAN12 IMSA 0 AUTPS 1 0 2 6553 6553500 7 7

F1=Help F3=Exit F4=Showlog F6=Expand F9=Retrieve F12=
Doppelt klicken, um IBM Personal Communications zu star
```

Display formatted by SPOC from
XML response from OM.

Note that the command entry line has been cleared, but the command text is shown in the response section.

This screen is an example of a command response which does not fit a single screen.

The

+ and > signs indicate more data below (+) and to the right (>).

Some IMSplex command responses will have key columns that do not scroll horizontally and will always remain on the screen, so the user can have a point of reference. The columns are defined as keys by the XML. In the example, the TranCode and MbrName values will not scroll horizontally.

Type1 Command and response

[illegible]

The command response for IMS Type 1 CMDs is in a sequential format. At the top: execution statistics and information.

Below:

Messages produced by the IMS command.

The text response is in the same format as if issued directly from some IMS system or the z/OS console. The text is prefixed by the member name. Information from each member is grouped together.

IMSplex commands might have log information, too, if there were some messages to display. For example, one system might say no resources found while other systems provide valid resource information. The no resources indication would appear in the log.