SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

An Introduction to IMS Transaction Manager *AND*… Why the Future is Brighter Than Ever

Suzie Wendler
wendler@us.ibm.com

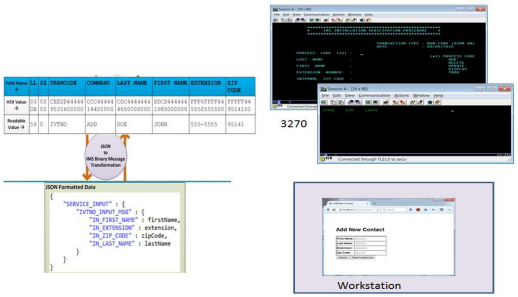
Being part of the project that put the man on the moon was just the beginning for the IMS product….

Since then, IMS became and continues to be an integral part of many business solutions and enterprise environments.

This session explains the architectural innovations that not only leverage what the z platform provides but also how technology advancements have provided the impetus for ongoing solutions for today and the future. Additionally, this session focuses on the transaction manager component and how transactions are processed. A separate session details the IMS database manager.

IMS Architecture

- Leverages a continuing investment
  - With minimal to no changes to the applications



Original Environment
- S/360 and OS/360
- Machines speeds: 200K to 800K instructions/second
- Machine memory – 256K to 512K- all memory was real
- 2311 Disk - 200 cylinders, 5 tracks/cylinder
  - 7,250,000 bytes per volume
  - 4GB would have required about 600 volumes!

First we put a man on the moon

And evolved over the years…. To today

Compiled programs written in accordance with the System/360 Principles of Operation in the 1960's ***continue to execute today***
→ and be accessed from the web and the cloud: mobile devices and API technology which did not exist when the product was created

IBM z and z/OS
>130K transactions/second on a single IMS
>265B transactions/day through IMS
>100M transactions/day on a single IMS system

IMS: Composed of 2 Products

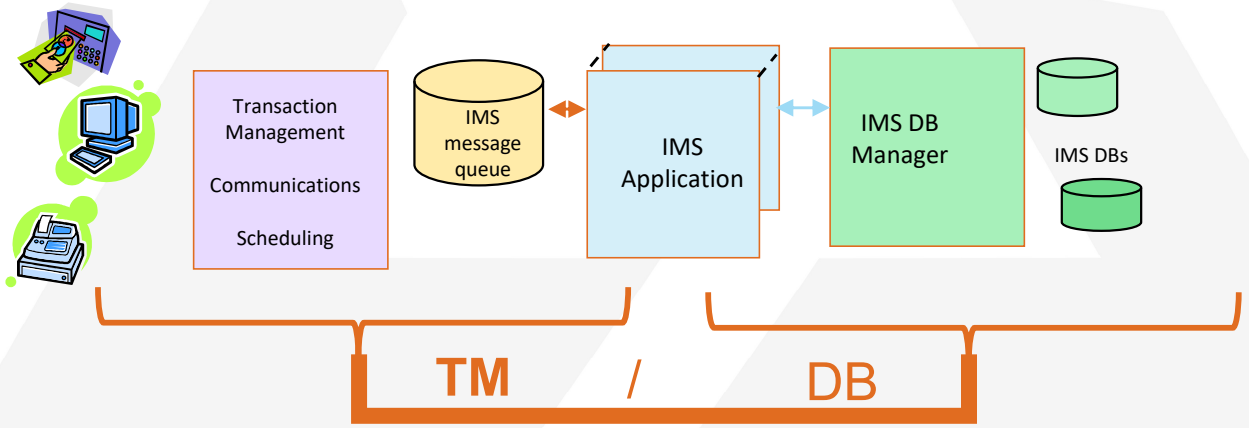
IMS DB: a Database Manager *(discussed in a different session)*

- hierarchical data structures

IMS TM: a Transaction Manager *(our focus in this session)*

- Used to be called IMS DC (data communications)
  - but was changed to TM because it provides more than just communications
    - > it also provide the processing environment for transactions

*So… **IMS DB/DC** is just the older name for **IMS TM/DB***



How?

IMS has allowed users to grow their application/environment in lieu of forcing major conversions … ***AND continues to do so with new functions and enhancements***

Application compatibility

Programs written decades ago still work - IMS does not require recompiles for new IMS releases or even when the communication mechanism to invoke the application changes.

Database compatibility

Databases do not require upgrades for new IMS releases

Database definitions outside of programs

Programming interface

DL/I calls used for db access - program not aware of data sets and physical characteristics of data

DL/I calls used for messages - program independent of communication protocols

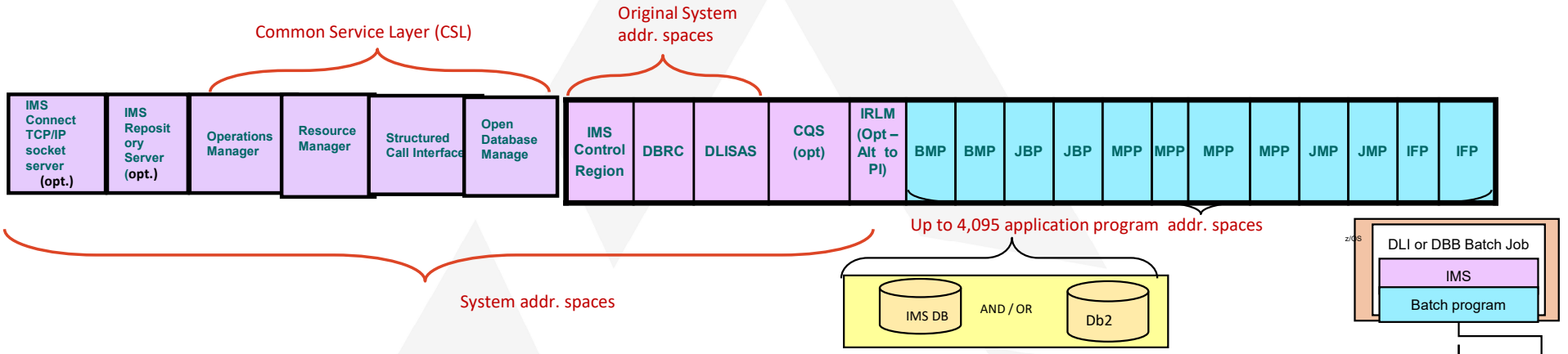
Additional support for JDBC/SQL access which hides the DI/I interface

Multi-region online system

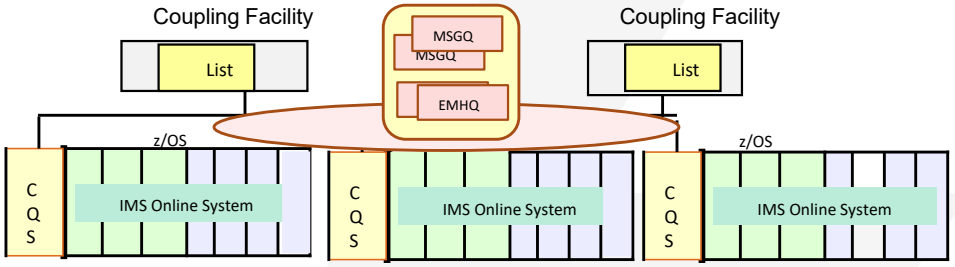
Failure isolation

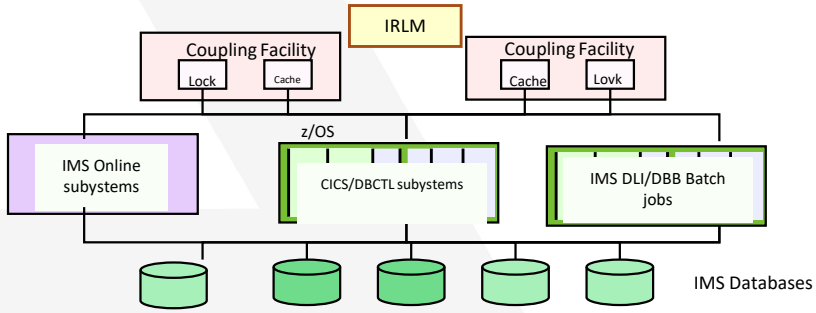
IMS Architecture Environment

- Single system environments



- And Multiple systems with parallel sysplex technology





Additionally…IMS keeps track of activity: IMS Logging

- IMS online systems log all significant activities
  - Provides for restart of failed IMS systems
  - Provides for recovery of databases

RECON - Recovery Control Data Set – tracks which logs are in use

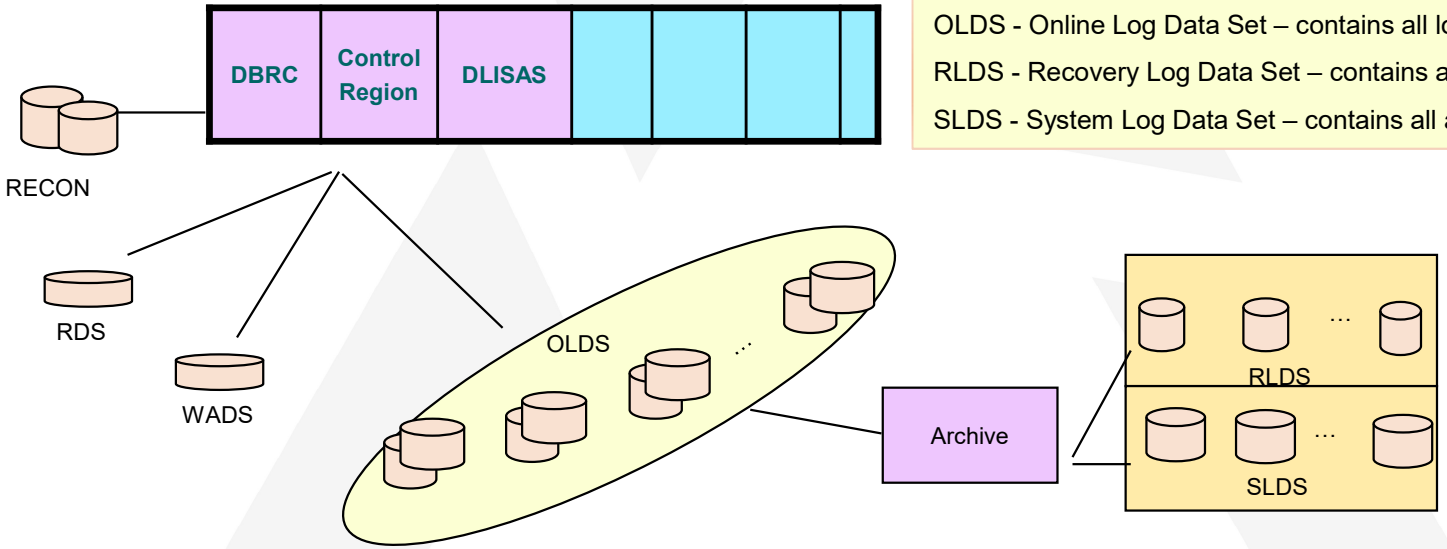
RDS - Restart Data Set – tracks which OLDS records are needed for recovery

WADS - Write Ahead Data Set

OLDS - Online Log Data Set – contains all log records and used round-robin

RLDS - Recovery Log Data Set – contains archived DB recovery log records

SLDS - System Log Data Set – contains all archived log records



The WADS contains a copy of committed log records that are in OLDS buffers, but that have not yet been written to the OLDS.

In order to maximize log efficiency, IMS uses a log write-ahead function to write partially filled blocks to the WADS (rather than the OLDS). IMS continually reuses WADS space after writing the appropriate log data to the OLDS. The log write-ahead function ensures that all log records are on the log before IMS writes changes to a database.

IMS Logging …

LOGS are used for database Recovery

- IMS logs all database updates
  - Provides utilities e.g., Image Copy utility; Change Accumulation (CA) utility which sorts log records and discards older updates to shorten recovery time; Database Reovery utiity to read Image Copy, CA, and log records to restore database data set; etc.

LOGS are used to track transaction processing events

- IMS logs all activites in the life of a transaction
  - Keeps track of messages, application events and statistics, queue manager information, etc.

LOGS are used for System Recovery

- IMS system logs significant changes
  - Status changes for resources, Commands, Database changes, IMS TM messages
- IMS writes periodic system checkpoints to log
  - Statuses of resources
- Restarts of the IMS System
  - Normal restart – to restore system from termination checkpoint
  - Emergency restart
    - IMS determines which logs to read
    - Reads system checkpoint and log records following system checkpoint
    - Backs out in-flight database and message updates
    - Restores statuses of resources

Log records are used by IMS utilities and tools to track what has gone on in the system

- **For example: tracking a transactions can be done using:**
  - 01 Input message into IMS
  - 35 Enqueue the message to the message queue
  - 08 Application scheduled
  - 31 Application retrieves message
  - …..5xDATABASE records
  - 56 Two phase commit
  - 07 Application termination (statistics – often used for auditing, billing)
  - 33 Queue manager released a record
  - 35 Dequeue the message
  - 03 Output message send from IMS

To see contents (DSECTs) of log records, assemble ILOGREC macro

ILOGREC RECID=01 (shows 01 log record mapping)

ILOGREC RECID=ALL (shows mapping of all the log records)

NOTE: IMS uses its own logger and log data sets and NOT SMF

IMS Application Environment - Evolution

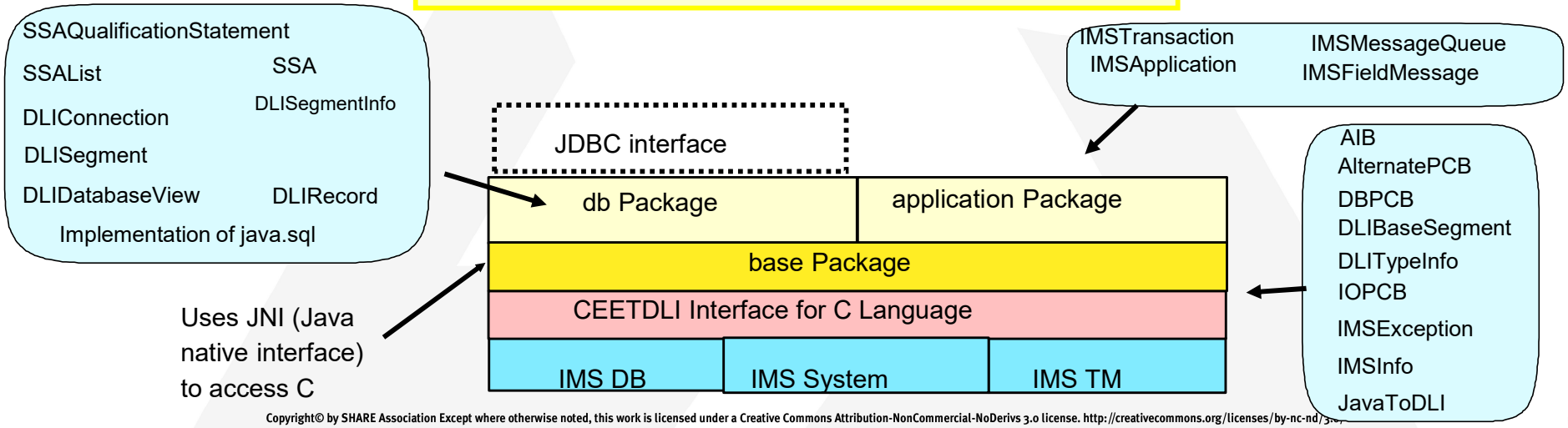
- Application programs run in separate address spaces (dependent regions)
  - Message processing program (MPP) regions (IMS TM online transactions)
  - Java message program (JMP) regions (IMS TM online java transactions)
  - IMS Fast Path (IFP) regions (IMS TM online transactions)
  - Batch message program (BMP) regions
    - *Special type of batch job that attaches to the online system and accesses the same resources (message queues and databases)*
  - Java batch program (JBP) regions – java version of BMP

- **IMS supports applications written in:**
  - COBOL, PL/I, Assembler, C/C++, Pascal, Ada, REXX … and Java
  - IMS TM messages are accessed with DL/I calls
    - Java programs use IMS-supplied Java classes
  - IMS databases are accessed with DL/I calls
  - Java programs may use JDBC which is converted to DL/I calls
  - Cobol and Java can use the JNI in an MPR
  - A distributed java application may access IMS databases directly
    - Leveraging IMS-provided universal drivers and IMS Open DB support

Application Evolution - IMS Java

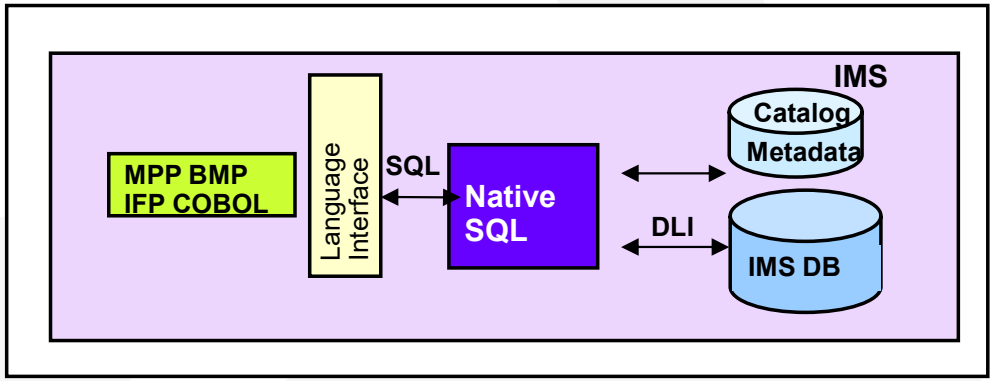
- IMS Java
  - JVMs are launched in dependent regions

Java program uses the APIs that are provided
- application Package classes to
  - initialize and begin the program
  - get the input message from the message queue
  - put the output message on the message queue
  - commit
- JDBC interface or db Package classes to
  - access the IMS databases



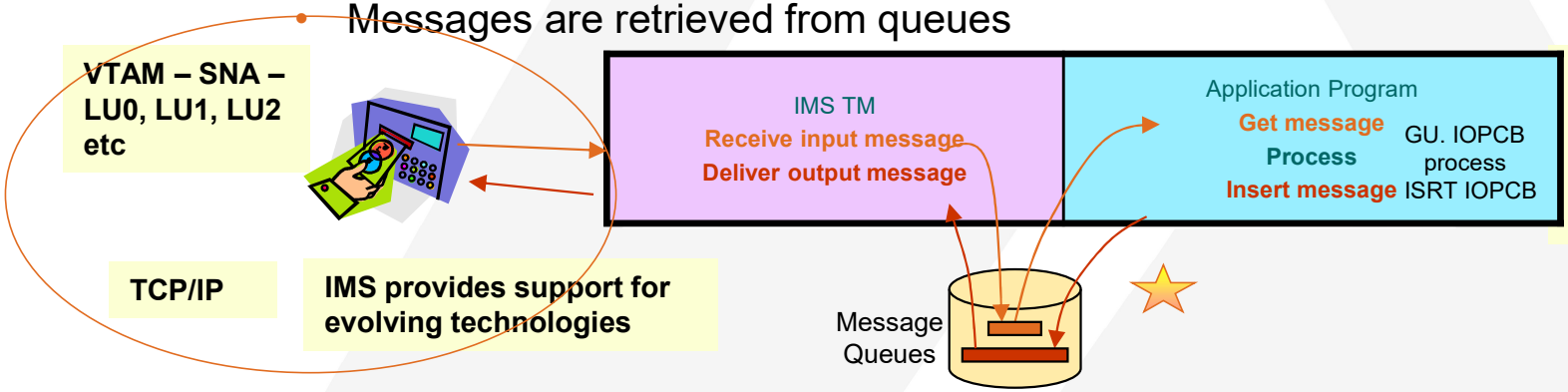
Application Evolution …

- And even… IMS Native SQL support beginning with COBOL 5.1+
  - Provides standard SQL keywords to easily access IMS data
    - SELECT, INSERT, UPDATE, DELETE
  - Uses Dynamic SQL programming model
  - Converts SQL statements to DL/I calls
  - Supports a subset of SQL keywords
  - Uses database metadata in IMS Catalog



A Closer look at Transaction Management

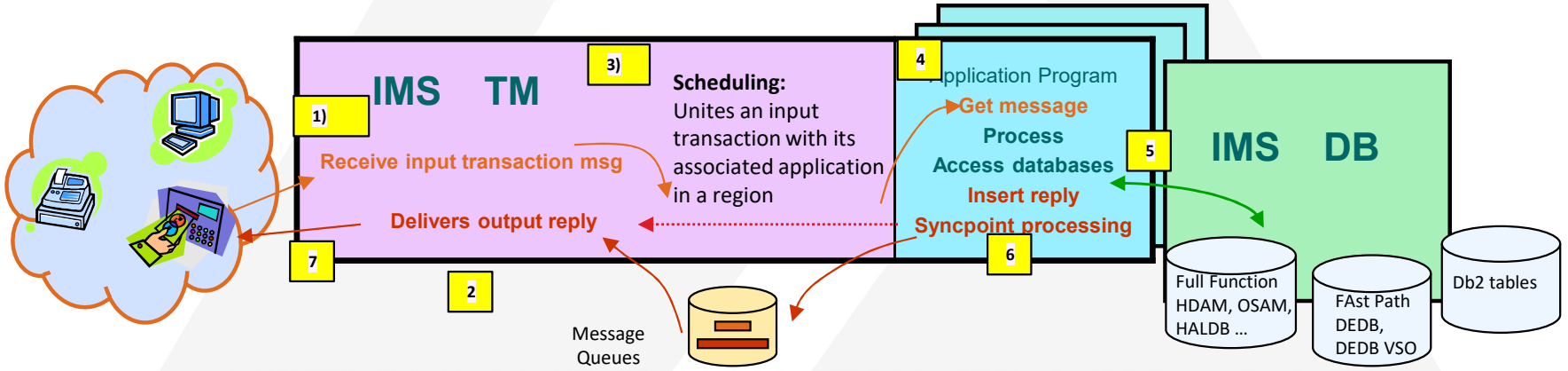
- Transaction management (IMS TM component)
  - IMS TM receives input messages from terminals or other systems
    - Messages are placed on message queues
      - *Note: IMS message queues were critical to the design from the beginning*
    - Input messages cause scheduling of programs
  - Application programs process input messages
    - Application programs get input messages from queues
    - Application programs insert output message to queues
  - IMS TM delivers output messages
    - Messages are retrieved from queues



A Closer Look – Message Processing

- **IMS Full Function message flow**

1. IMS TM receives an input message
2. The message is placed on the IMS message queue
3. The message is scheduled for an application program in a region
4. Scheduling end to first DL/I call
5. Program processes the message including access to database(s)
6. Syncpoint
7. Delivery of the output reply



A Closer Look ...

SHARE

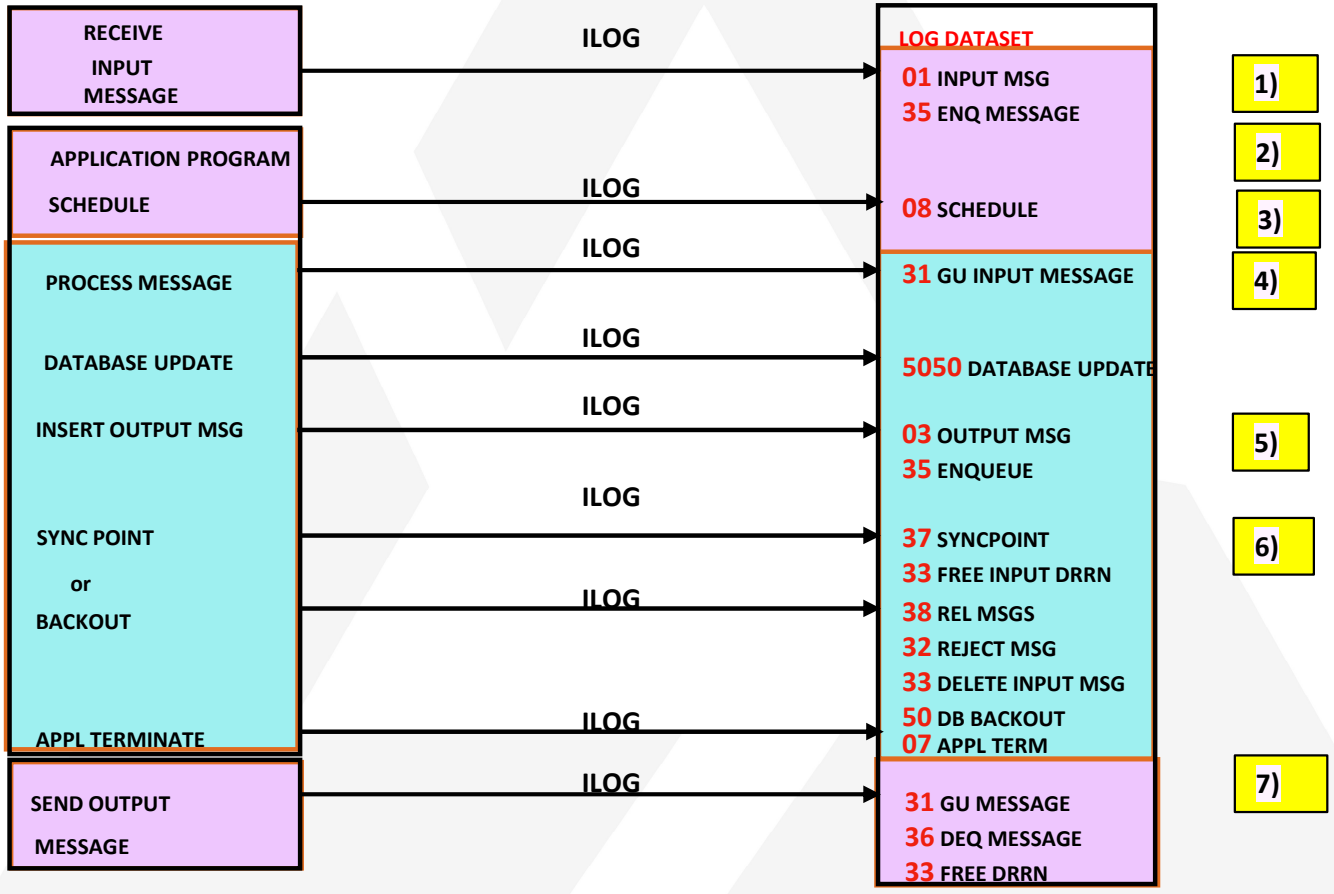
NCE

Full Function message flow

	Event	Activity	Pools/Lists (Examples)	Potential issues
1)	IMS Receives an input message	Editing routines, security, Exit routines	RECANY, CWAP HIOP,CIOP LUMP/C, MFP	Erratic transaction arrival rates with bad response times - possible pool shortages or selective dispatching
2)	Message is placed on the IMS message queue	Message is queue to the scheduler message block (SMB) for the transaction	QBUF	Possible mssage queue buffer (QBUF) shortage, message queue I/O bottlenecks
3)	Message scheduling	Schedule the message in an available dependent region	CSAPSB, DLIPSB, PSBWP, DMB, DBWP	Transaction queuing, pool intent failures, No regions available, scheduling options
4)	Scheduling-End to First DL/I call	Load programs, subroutines, and initialize working storage	BLDL, LLA/VLF, Preload	Program load options, program initialization problems, could also show up as high cpu or elapsed times
5)	Program execution	Application pgm is invoked and DL/I calls performed	QBUF, OSAM/VSAM buffer pools	High elapsed times as a result of application code, database calls/buffering, I/O or system issues , DL/I IWAIT or Non-IWAIT
6)	Syncpoint	Phase 1 and Phase 2 Commit	OSAM/VSAM buffer pools	High elapsed times as a result of WADS, OLDS or I/O subsystem delays
7)	Deliver the output reply	Send output to destination	QBUF, MFP, CIOP, HIOP	Pool shortage or network delays

A Closer Look ...

- **Understanding What The Transaction Has Done – Log Records**



A Closer Look ...

- **IMS also has Fast Path Messages**

Expedited Message Handler (EMH) - Provides an alternative message queuing and handling capability

- IMS determines if the message is to be processed as a Fast Path message
- Assigns a routing code (RTCODE) which causes the message to be queued to a balancing group (BALG) which is the equivalent of a full function SMB (transaction block)
- Uses EMH buffers instead of message queues

Why use Fast Path?

- High performance with lower CPU cost

Restrictions – Limited to processing:

- Wait-for-input only
- Single segment input and output messages
- Response mode transactions only
- Non-conversational transactions only

IMS Fast Path Message Flow

1. IMS TM receives an input message
2. Fast Path EMH queuing
3. Fast Path EMH scheduling
4. Program processes the message including access to database(s)
5. Syncpoint
6. Delivery of the output reply

A Closer Look ...

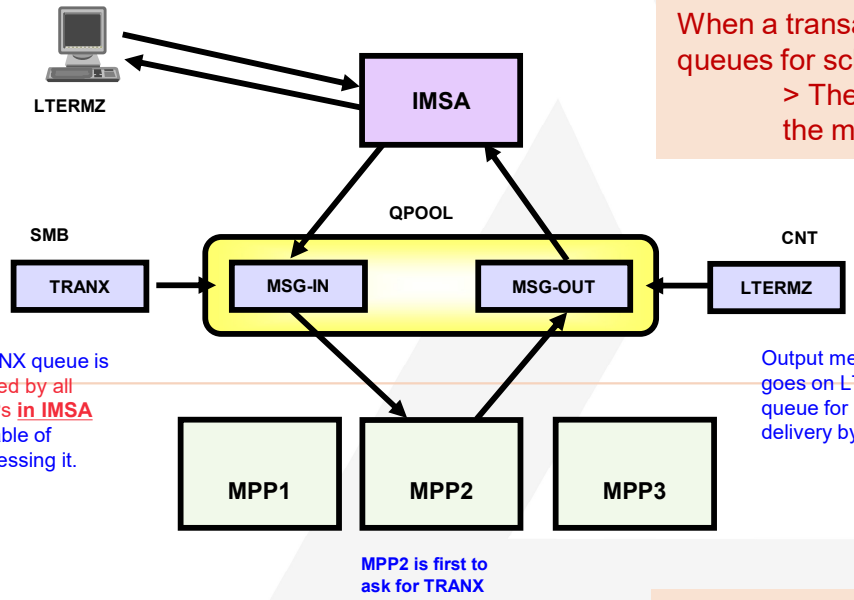
- **IMS also has Fast Path Messages …**

IMS Fast Path message flow		
Event	Activity	Potential issues
IMS Receives an input message	Editing routines, security, Exit routines	Erratic transaction arrival rates with bad response times - possible pool shortages or selective dispatching
Fast Path expedited message queuing	Determine if FP potential or exclusive	EMHB pool shortage
Fast Path EMH scheduling	First-In-First-Out scheduling by Balancing Group (BALG)	Queuing on BALG
Program execution	Application pgm is invoked and DL/I calls performed	High elapsed times as a result of application code, database calls/buffering, I/O or system issues
Syncpoint	Phase 1 and Phase 2 Commit	Output thread (OTHR) shortage
Deliver the output reply	Send output to destination	EMHB pool shortage or network delays

More information : https://ibm.biz/BdbpCi

A Closer Look - Scheduling

- IMS message queues were critical to the design from the beginning…
  - *Separate the application from having to understand the communication method*



When a transaction arrives, it is placed in the IMS message queues for scheduling and processing
> The message queue pool acts as a buffer to the message queue data sets.

TRANX queue is shared by all MPPs **in IMSA** capable of processing it.

Output message goes on LTERMZ queue for delivery by IMS.

Load balancing is achieved by "sharing" these queues across dependent regions within that IMS
> Whichever dependent region becomes available to process a transaction would be the one to get it off the queue

The system programmer's role is simply to be sure that the correct number of dependent regions are scheduled.

Transaction Management

Trancode is defined to IMS
Sysdef: TRANSACT macro with a CLASS number

IMS dependent region address space
Specifies: which classes can run in it

Input message arrives

Transaction code is typically in the first 8 bytes of message

LLzz | TRANCODE | data…..

IMS places message on queue for the transaction code

LLzz | data…..

Messages can be single or multiple-segment (each segment can be up to 32K)

IMS dependent region (*MPR – message processing region*) ***asks*** for work

Region can process any transaction code in its classes  (or be dedicated as a wait-for-input)

Each transaction belongs to a class

Highest priority transaction available for region's classes is given to the region

IMS dependent region loads scheduled program and gives it control

Program may process multiple input messages; insert output messages to queue

When program terminates, dependent region asks for work

Output message

Placed on queue for output destination

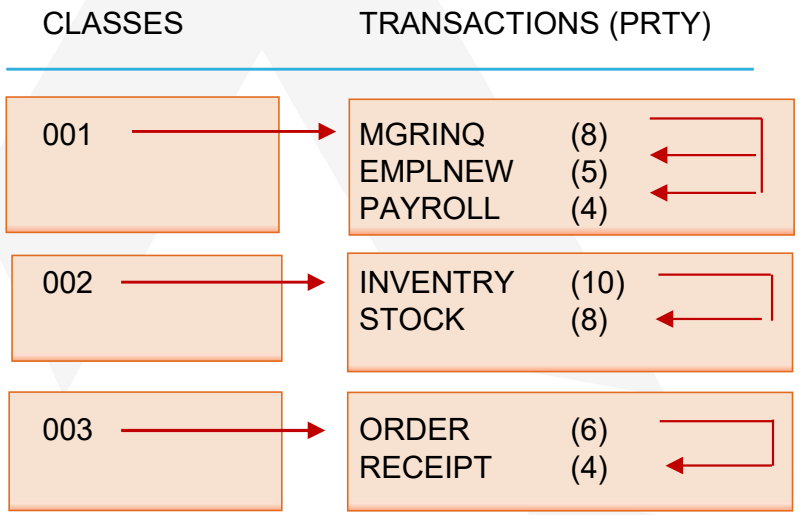
LLzz | data…..

Delivered to output destination by IMS

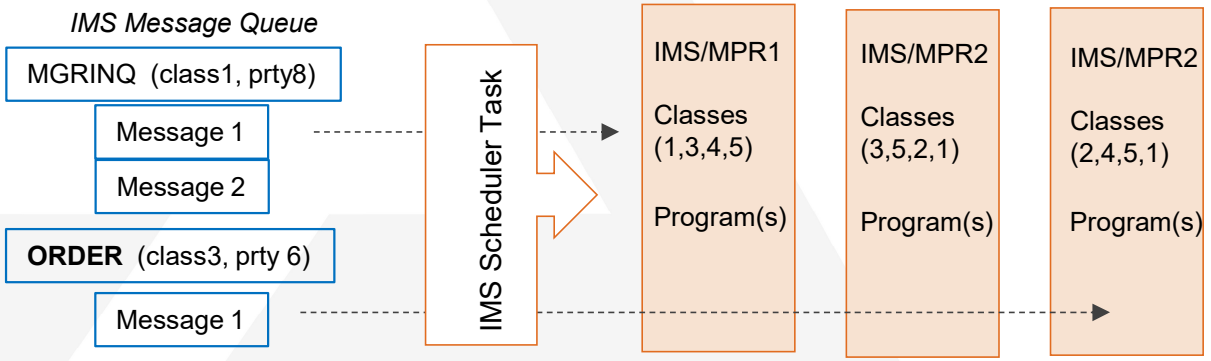
LLzz | data…..

Transaction Management …

- Transactions have attributes
  - Name (<=8 characters)
  - Class (1-999)
  - Priority (0-14)
  - …



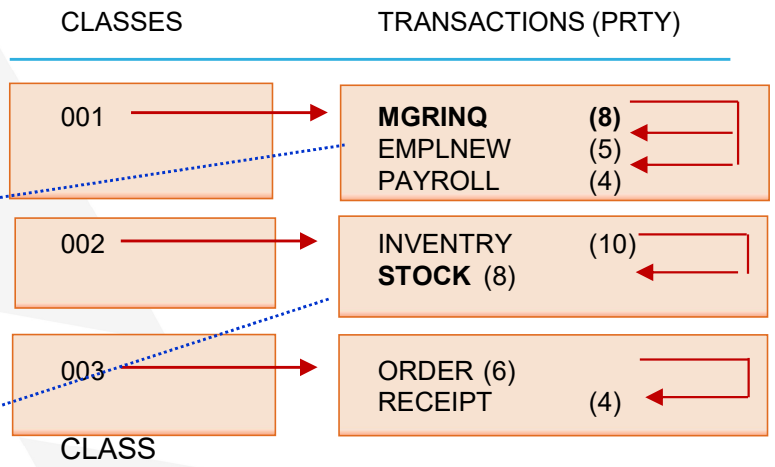
- The IMS Scheduler selects an instance of a transaction message for processing by an MPR based on the Class and Priority of the transaction.



Transaction Management …

- **Attributes are assigned as part of the definition**
  - **Transactions are associated with programs**

```
APPLCTN PSB=INQPGM,
PGMTYPE=(TP,,1)
TRANSACT CODE=MGRINQ,
PRTY=(8,10,20),
PROCLIM=20
..
APPLCTN PSB=UPDSTK,
PGMTYPE=(TP,,7)
TRANSACT CODE=STOCK,
PRTY=(8,12,4),
MSGTYPE=(,,2)
```



OVERRIDES CLASS in APPLCTN macro

- Definitions can be created and modified either through a ***sysgen batch process*** or ***dynamically through online (Type 2) commands***

Transaction Management

IMS scheduling - determines which region is available to process the message

Transactions

Are defined with a **class** value and whether it can be **parallel scheduled** and its **Priority** with respect to other messages

Can specify that it is **Wait-For-Input (WFI)** and needs a dedicated region(s)

Regions

Can specify up to 4 classes and can process work on behalf of multiple transactions

Can be defined as **Pseudo-Wait-For-Input (PWFI)** which takes effect when there is no work to do

The region remains scheduled until another input message appears, anticipating that the next message is for the same transaction that was just run and avoiding unnecessary application program termination and rescheduling

Quick reschedule is a function of the scheduler when IMS determines there ***is more of the same work to do*** and overrides the transaction processing limit (PROCLIM) defined for a physical schedule in order to eliminate unnecessary rescheduling and reloading of application programs

Algorithm when the processing limit is reached

- If any equal or higher priority transactions are queued, IMS terminates the application program. The region becomes available for another program to be scheduled into its storage. IMS uses the scheduling algorithm to choose the program to schedule.
- If no equal or higher priority transactions are queued and messages are still queued for the current application program, the region goes through ***quick reschedule*** and returns the next message to the application program.
- If equal priority transactions are enqueued, IMS allows the transaction to quick reschedule if the PROCLIM value has not been reached. If the PROCLIM value has been reached, IMS disallows the quick reschedule and processes the other transactions.

Transaction Management

And then there is Fast Path

DB component discussed in a later presentation on IMS DB

Database Component: DEDB (Data Entry Data Base)

Transaction Manager component

Fast Path transactions run in an IMS Fast Path (IFP) region and not an MPR

Expedited Message Handler (EMH):

Simplified message flow - Bypasses normal IMS/TM message queue and message queue handler

Simplified Scheduling versus using the IMS scheduler

TM/DB and DCCTL systems only - not available for DBCTL

Goal: provide the highest possible level of performance for database and transaction processing

Used for simple transactions when data communication requirements are for a high transaction volume with rapid database updates and inquiries

Examples of use: teller transactions in banking and point-of-sale transactions (inventory update) in retail marketing

Higher level of performance is achieved by using different logging protocols, providing somewhat reduced functionality, and by imposing restrictions on its use for DB and TM processing as compared to what is provided for 'full function' processing in MPR regions

Transaction Management

And then there is Fast Path …

Expedited Message Handler (EMH)…

Each IFP (Interactive Fast Path) region processes only one program and associated transactions

Similar to a WFI Message Processing Region

Once a program is scheduled in an IFP Region, it will wait for new instances of transactions to arrive and be passed by EMH to that IFP

transactions are selected on a first-in/first out basis as they arrive

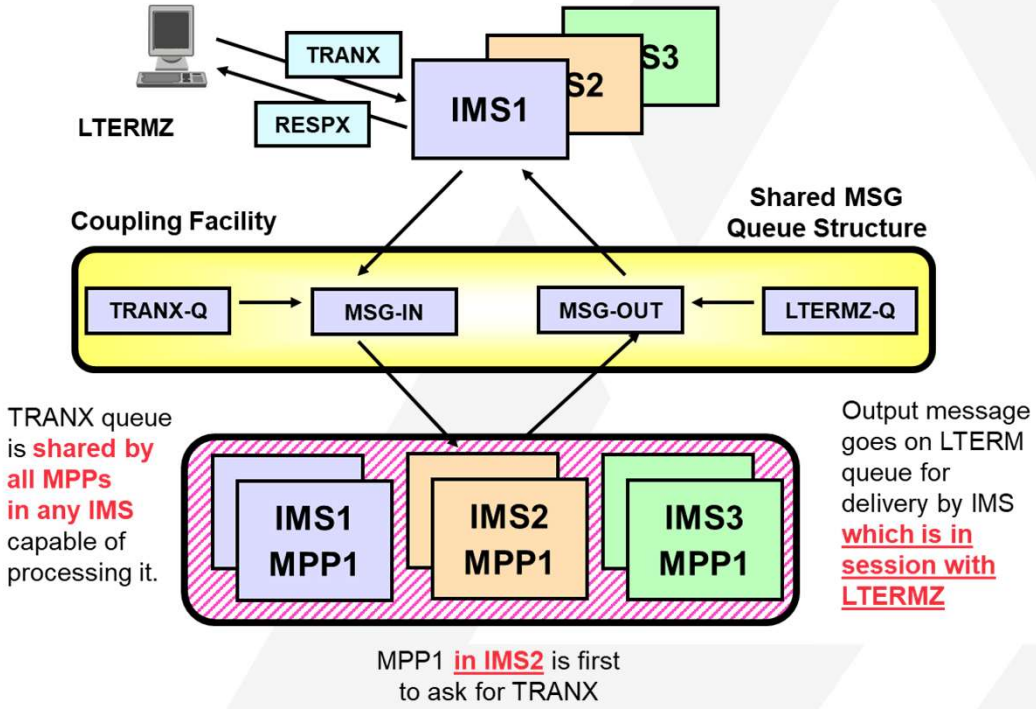
- Additional CPU resources and time are saved by eliminating the processing performed by the IMS Queue Manager that is associated with the queueing and dequeueing of messages in the IMS Message Queue
- Messages can only be single-segment

IMS systems can process both full function and fast path transactions

Support both types of scheduling as well as MPR and IFP regions

And with Parallel Sysplex…

Shared Queues…



1.IMS systems register interest in those queues for which they are able to process messages.
- Multiple systems processing the same tran would need to implement datasharing

2.When an IMS receives a message and places it on the shared queue, all IMS systems that have registered interest in that queue are notified.
- Local-first processing allows the system that receives the message to immediately process it if able

3.One IMS retrieves the message and processes it.

4.The IMS that processes the message places a response on the queue.

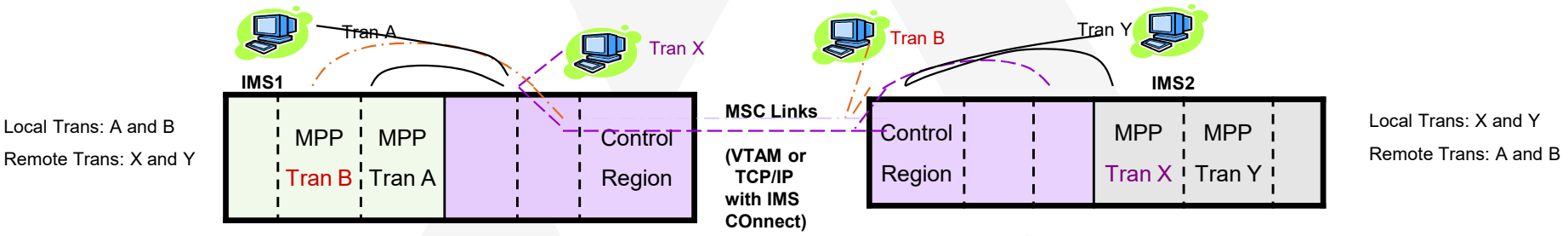
5.The IMS that submitted the original message is notified that the response message was placed on the queue.

6.The IMS that submitted the original message sends the response message to the originating terminal.

But Even Without a Sysplex

Multiple Systems Coupling (MSC) – *Supports growth by connecting IMS systems in the same of different locations*

- Allows transparent end-user access to transactions in any of the systems in the MSC network



- Through seamless workload routing – no changes to the application programs
  - Based on definitions, IMS routes and controls message traffic between connected IMS systems
  - Transactions may be defined as remote (processed by another IMS) with responses sent back to the sending IMS/terminal
- Communication between 2 or more (up to 2036) IMS systems can run on any combination of supported operating systems
  - Supported link types: channel-to-channel (CTC), memory-to-memory (MTM), VTAM, TCP/IP
    - No restriction on VTAM or TCP/IP link distances
- With IMS DRD (Dynamic Resource Definition) support - dynamic MSC definitions can be enabled in an online system

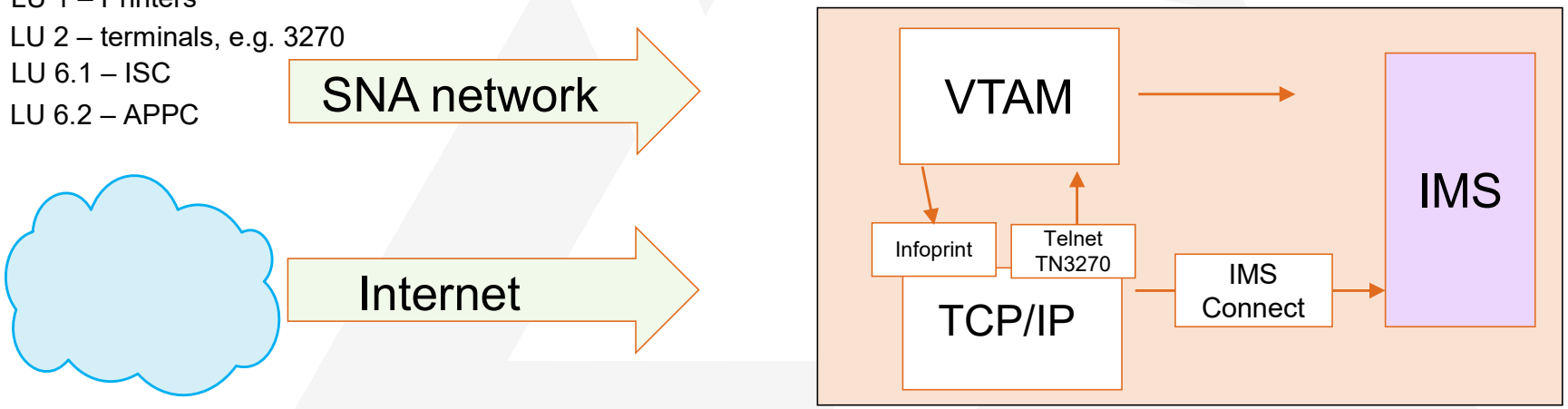
And Then … The Communications Evolution

Originally, IMS transactions were accessed via SNA VTAM

Even today, IMS is still a VTAM application

But… over time, access from TCP/IP networks and the Internet became essential

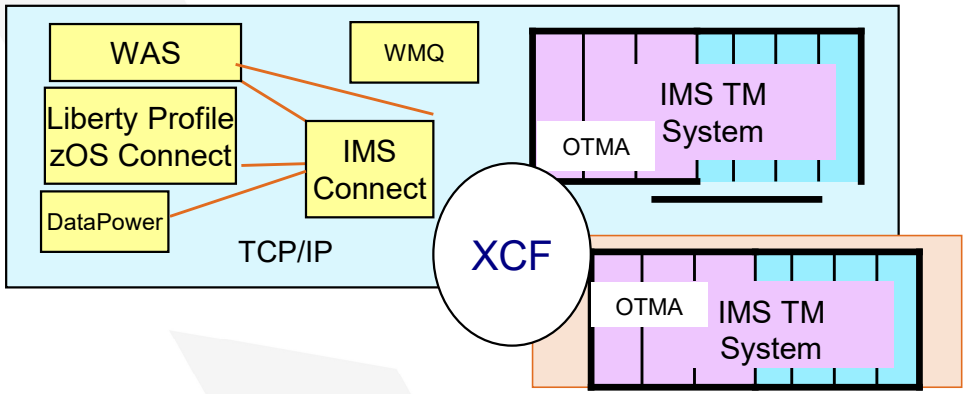
LU 0 – undefined (MSC)
LU 1 – Printers
LU 2 – terminals, e.g. 3270
LU 6.1 – ISC
LU 6.2 – APPC



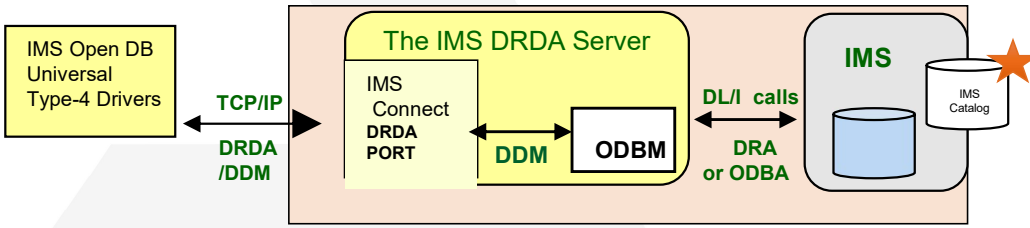
Communications Evolution …

- To support evolving technologies, the architecture expanded with:

z/OS XCF technology – allows address space communication on the same or different LPARs



- **OTMA (Open Transaction Manager Access)**
  - Provides standardization of access to IMS transactions from non-SNA interactions
    - Clients: IMS Connect, WMQ, WAS, etc
    - .
- **IMS Connect (IMS TCP/IP Socket server)**
  - Opens up standardized access to IMS transactions and databases from TCP/IP clients
- **ODBM (Open DataBase Manager)**
  - Works with IMS Connect to provide distributed (DRDA) access to IMS DB



Communications Evolution – IMS Connect

IMS Connect is a TCP/IP Server which supports TCP/IP communications between:

- IMS - CICS for ISC sessions (ISC/TCPIP)
- IMS - IMS for MSC sessions (MSC/TCPIP)
- IMS - IMS for asynchronous program-program switches (IMS-IMS/TCPIP)

For Transactions, IMS Connect is an Open Transaction Manager Access (OTMA) client

- IMS Connect and OTMA can be on the same or different LPARs
- IMS Connect accepts input messages from and sends output messages to clients
  - Could be any TCP/IP Socket application - IMS Connect publishes its application protocol
  - Could be a Java Application, Bean, Applet, or Servlet
  - Supports the REST API interface through z/OS Connect

For Commands, IMS Connect provides Type-2 command supports

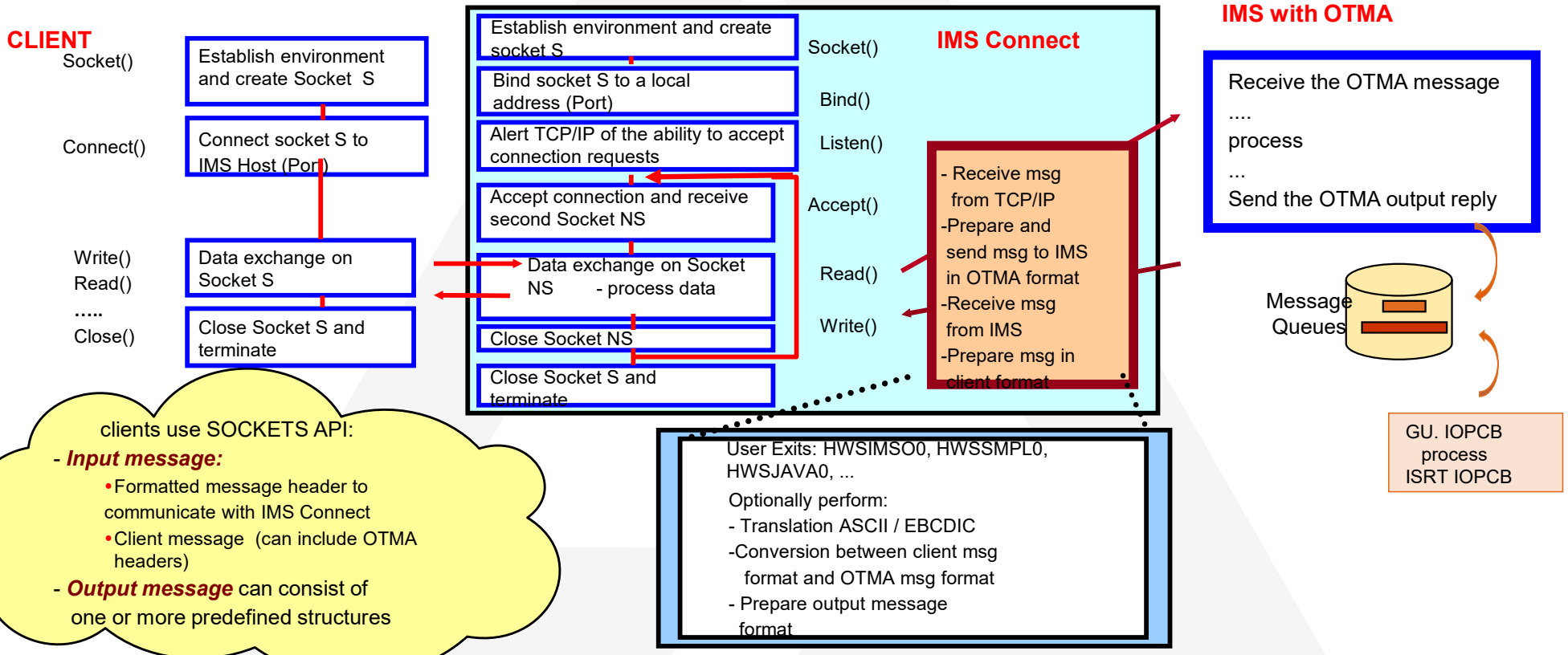
- Interfaces with Operations Manager (OM) via the Structured Call Interface (SCI)

For Databases, IMS Connect with Open Database Manager (ODBM) is a DRDA server

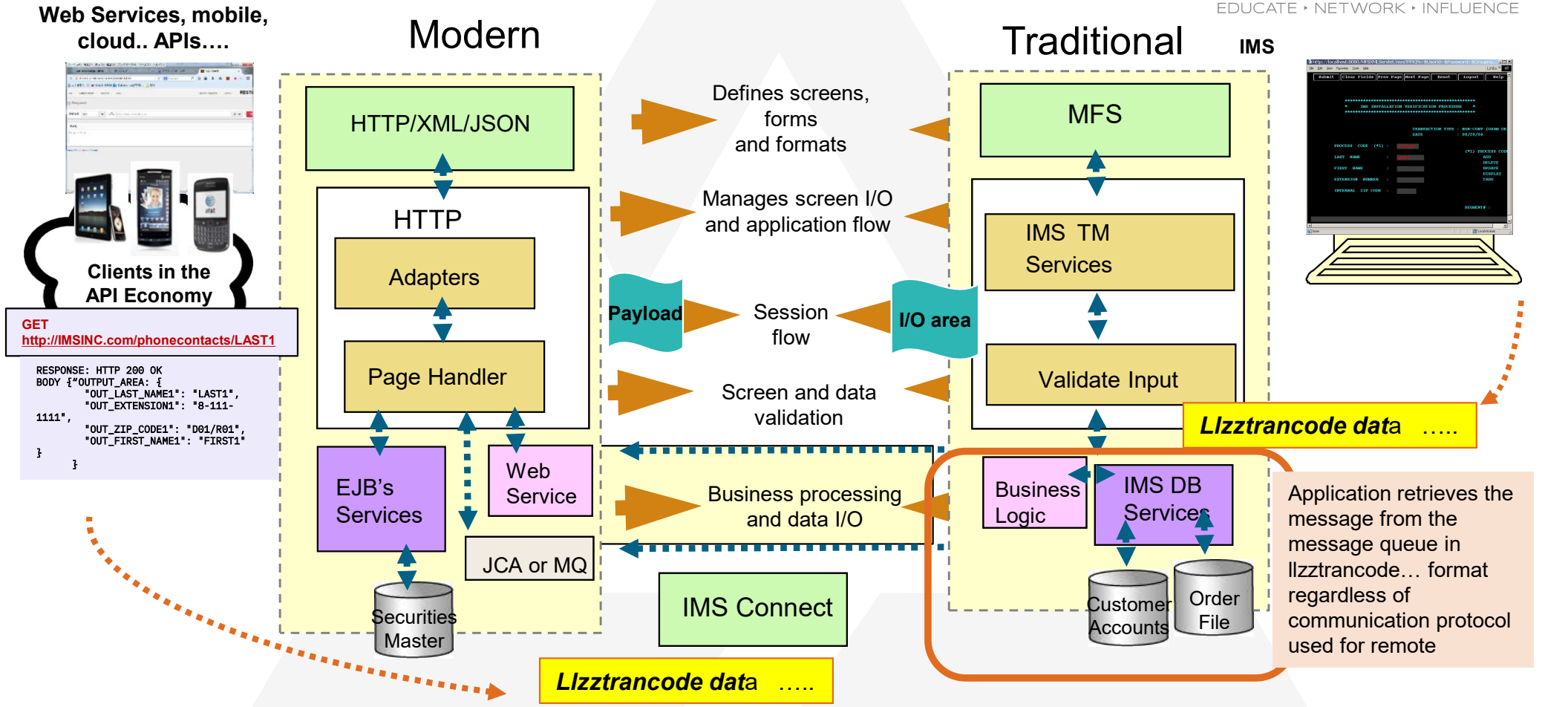
- Supports Open Database access from the IMS universal Drivers

Communications Evolution - IMS Connect

- For IMS Transactions --- IMS Connect protocol



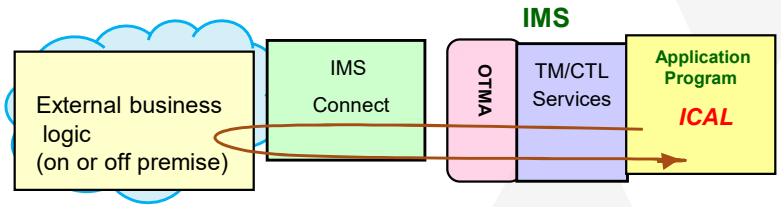
Handling Messaging – Traditional <-> Anything New



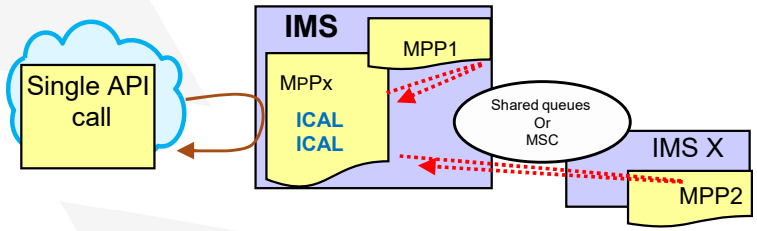
And… Continuing the Application Evolution and Extension

- **Positioning IMS to be an integrated partner with cloud environments**

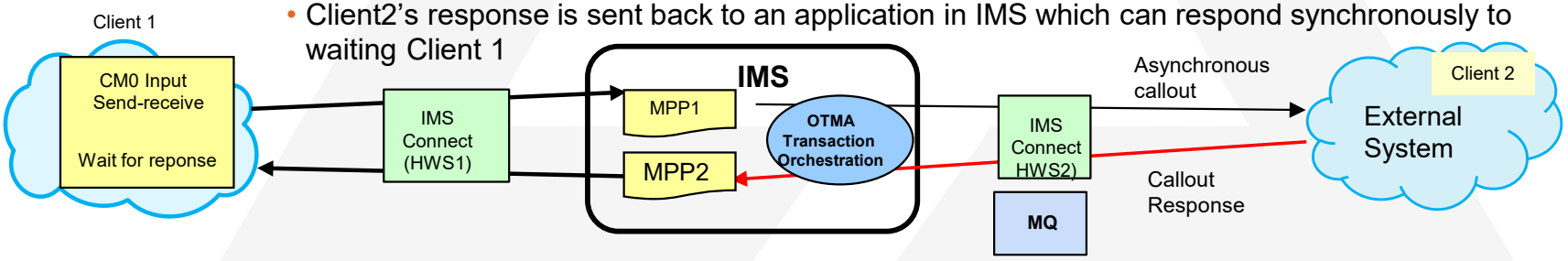
  - **Synchronous Callout**
    - DI/I ICAL – ability to synchronously call to an external program and wait for a reply



  - **Synchronous Program-to-Program Switch**
    - DI/I ICAL – ability to synchronously call another IMS program and wait for a reply
      - Can create a 'broker' app in IMS

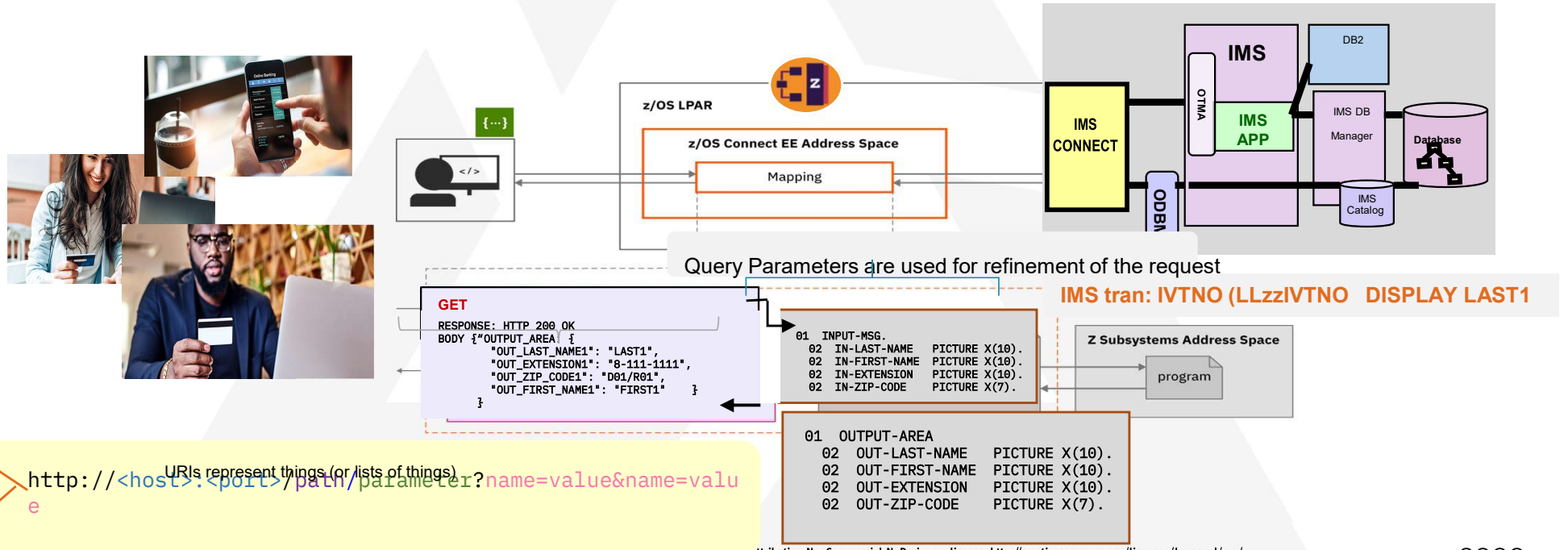


  - **Transaction Orchestration**
    - Allows a remote client (client 1) to wait for a reply from IMS
      - While the IMS application (MPP1) uses asynchronous protocols, e.g., ISRT ALTPCB to client 2
      - Client2's response is sent back to an application in IMS which can respond synchronously to waiting Client 1



Including Support For APIs

- **By leveraging z/OS Connect and IMS provider code**
  - Support for OpenAPI2 (Swagger 2.0) – IMS service provider, API requester, IMS DB service provider
  - Support for and OpenAPI3 (OAS3) – IMS provider, API requester



GET
POST
PUT
DELETE

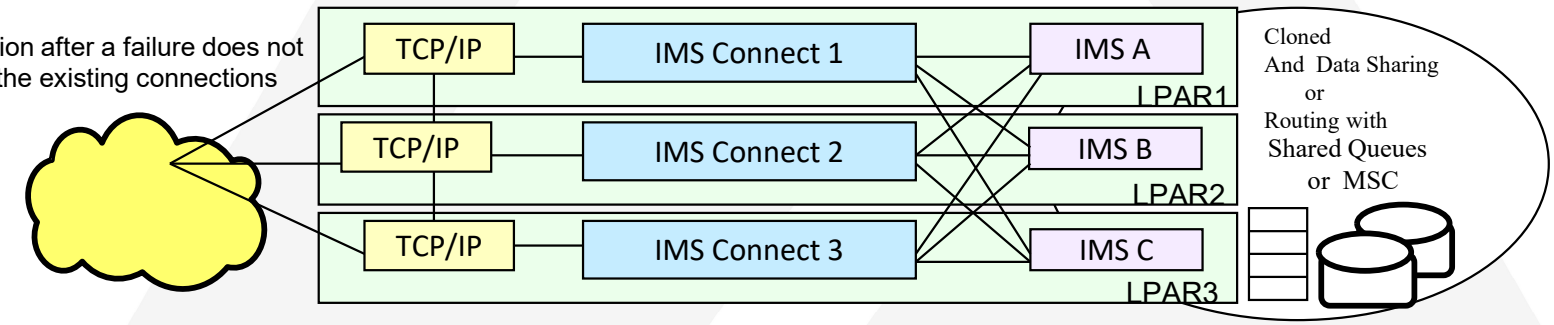
http://<host>:<port>/path/parameter?name=value&name=value

URIs represent things (or lists of things)

Along With High Availability

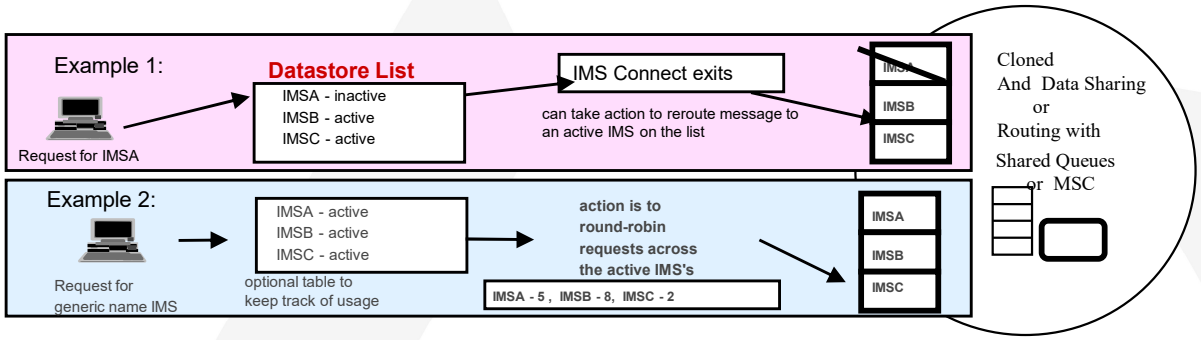
- **The use of mechanisms such as IP spraying, workload balancing, port sharing and sysplex distribution**
  - Allow a connection request to be routed to any of the available IMS Connect instances
    - Different versions of IMS Connect and IMS can coexist
  - Interfaces with Sysplex Distributor (SD) and Workload Manager (WLM) for workload balancing and failover
    - Allows WLM to know when resources are constrained or available
      - Minimizes the possibility that that SD assigns work that an IMS Connect cannot handle

Note:
- Reconnection after a failure does not rebalance the existing connections



Including IMS Connect <-> IMS

- **Balancing network connections does not always mean balancing workload for IMS**
  - So Additionally, IMS Connect provides mechanisms (exit routines) for rerouting of a request for load balancing or if an IMS is unavailable
    - Assumes the other IMS can handle the request



Tooling, e.g. IMS Connect Extensions (IMS CEX) enhances the functionality of IMS Connect for load balancing and rebalancing after a failure, routing, problem determination, etc.

- **Without having to write exit routines**
- **Also provides a journal for auditing and problem determination**

With a Continuing Communications Evolution

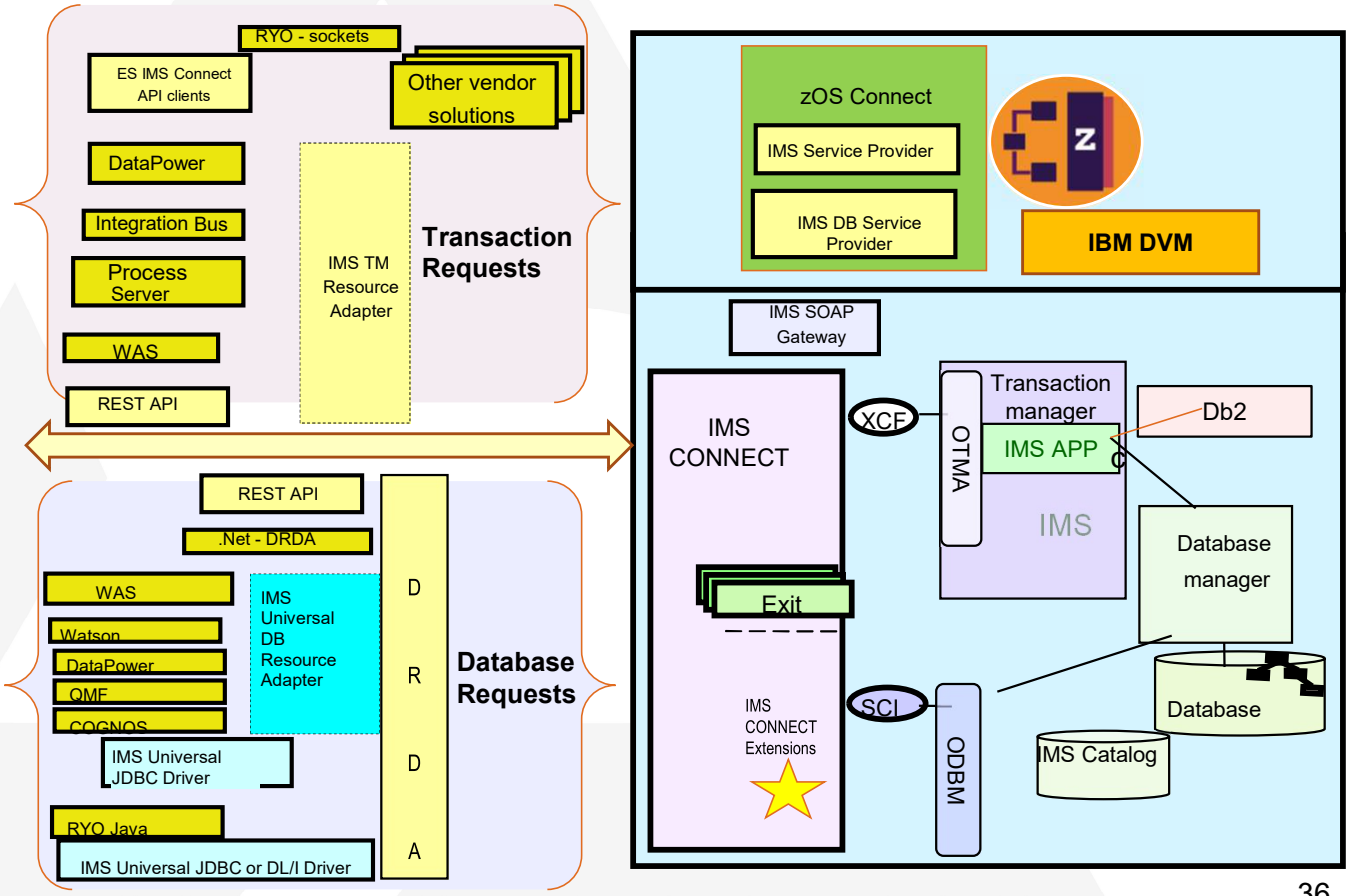
Resilient support for strategic solutions:

Cloud
- Quickly deploy new services
- Facilitate on premise provisioning of IMS resources w/o an outage

Mobile
- Scale due to increase loads
- Enhance information
- Maintain availability for 24x7 access

Analytics
- Easily add analytics solutions for IMS data

Bottom line: Ability to deploy strategic initiatives in IMS without impacting existing availability



So… In Summary

- IMS as a transaction manager
  - Multi-address space architecture
  - Processes messages
  - Schedules and manages programs
  - May be accessed via VTAM, TCP/IP, MQ,…

    … without having to modify the application unless new functionality is desired
  - Accesses IMS and Db2 databases
  - ***All with an architecture base that facilitates ever-evolving and expanding technologies***

That is why the future is brighter than ever !!!

Experience more with IBM

Visit us at the IBM Booth #113

After a full day of technical sessions, take a break with us!

Connect with our experts, snap a photo with the z17 Plexi or the latest Telum II, and get an up-close look at our Spyre Accelerator.

Come back each day for fresh topics and demos at our expert stations.

Think 2026

Join 5000+ senior business and technology leaders who are seizing the AI revolution to unlock unprecedented growth and productivity at **Think 2026**.

Find out more information using the QR code below.

IBM Digital Asset Haven

IBM Digital Asset Haven is the operational backbone for financial institutions and regulated enterprises entering the digital asset economy.

Find out more information using the QR code below.

Want to attend an in-person IBM z/OS Academy?

Learn, **Interact** and **Network** with IBMers and peers

May 5th- 7th, 2026	Fall 2026
IBM Tech Campus	IBM US
Ehningen, Germany	New York, USA

These **free** events are designed for early tenure z/OS system programmers (2-10 years), but all are welcome!

Training and presentations include topics on new z/OS capabilities, best practices, career tips, and **much more!**
Subscribe to the community page today to stay informed about future events!

Join our IBM Community: https://ibm.biz/zOSAcademy
Questions? Contact us at zOS.Academy.USA@us.ibm.com **or** zOS.Academy.Europe@de.ibm.com

Register now for Ehningen/ Germany:

Your feedback is important!

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

Submit a session evaluation for each session you

www.share.org/evaluation

40