Wildfire Workshop

Washington Systems Center
Technical Hands-On Workshops

IBM MQ for z/OS Overview and Terms

Dorothy Quincy
Technical Specialist
Dorothy.Quincy@us.ibm.com

IBM Washington Systems Center

MQ Overview Agenda

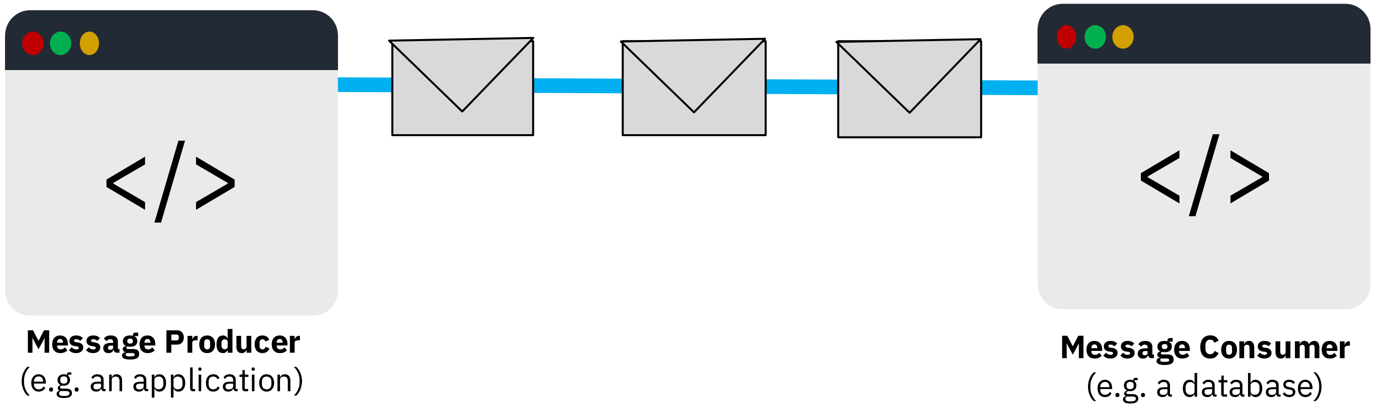
General MQ Terms

- Messaging
- Queueing
- Queue Managers
- Channels
- Publish/Subscribe

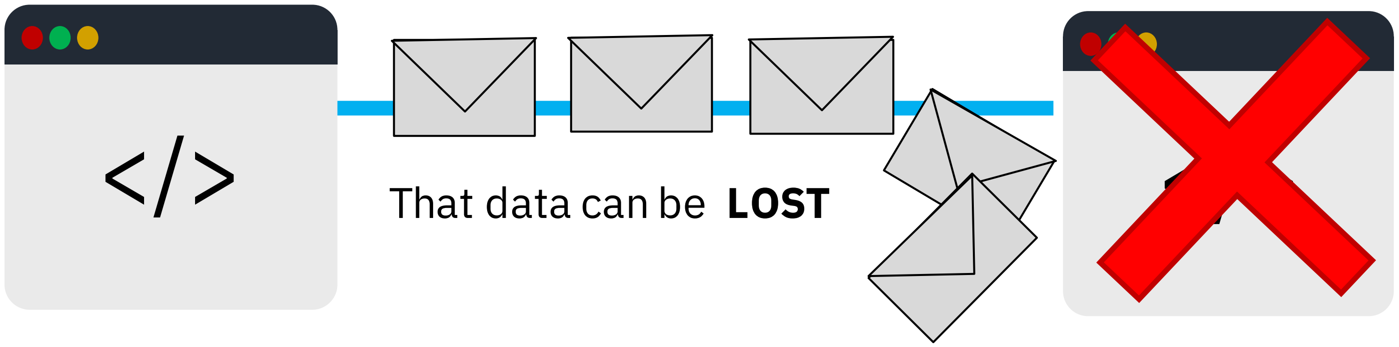
Configurations and Shared Queue Terms

- Client/Server Model
- QM Clustering
- Shared Queues
- List Structures and Coupling Facilities
- Intra-Group Queueing

Applications, services, systems etc. send data to each other.



But if there is a problem with infrastructure or the receiving application...



IBM MQ is *the* solution for business-critical messaging

The world depends on reliable, secure messaging and **85% of the fortune 100 depend on IBM MQ***

Your bank transfers complete without losing your money, with **all of the worlds top 50 banks using IBM MQ***

1 + 1 = 2

Simple

Scalable

Precise

Connected

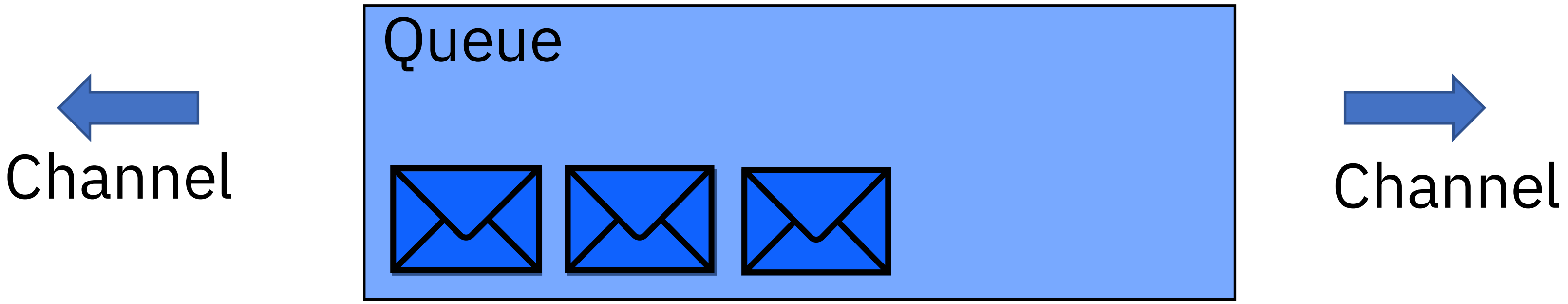
Reliable

Secure

* Sources:
http://www.relbanks.com/worlds-top-banks/assets
http://beta.fortune.com/fortune500/list/

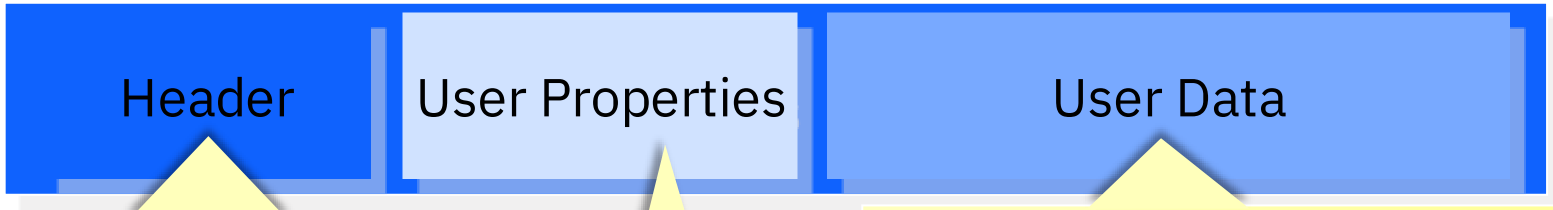
IBM MQ – Basic Terms

- Messages can be created from any source:
  - *Data, Messages, Events, Files, Web service requests / responses*
- Messages are moved asynchronously using <u>Queues</u>
- Queues are owned and managed by a <u>Queue Manager</u>
- Messages flow between queue managers across <u>Channels</u>



What is a Message?

Message = Header + User Properties + User Data



A Series of Message Attributes
Understood and augmented by the Queue Manager
- **Message Id**
- **Correlation Id**
- Message persistence
- Routing information
- Reply routing information
- Message priority
- **Message expiry**
- Message codepage/encoding
- Message format

....etc.

User Properties are Arbitrary properties
- For example, this is a “green” message

- Any sequence of bytes
- Private to the sending and receiving programs
- Not meaningful to the Queue Manager

Message persistence

A key attribute of a message is its persistence. A message is either persistent or non-persistent. This attribute tells the Queue Manager how important the message is.

Persistent

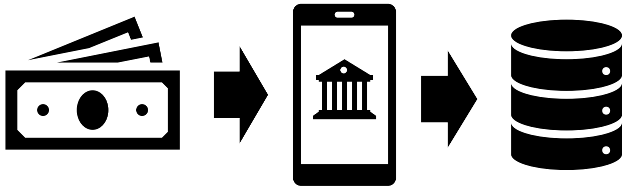
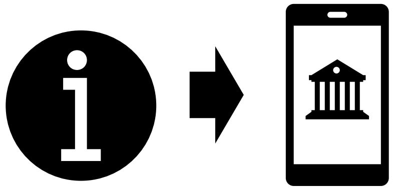
- Persistent messages are **logged** to the MQ log files (DASD).
- The Queue Manager will ensure that the messages are **recovered** in the case of a system crash or network failure.
- These messages are delivered once and only once to the receiving applications.

Non-persistent

- The messages are identified by the application as **non-critical**.
- The Queue Manager will make every effort to deliver these messages but since they are not necessarily written to disk, they will be lost in the case of a system crash or network failure.
- Clearly with no disk IO involved these messages are much faster (and cheaper) than persistent ones.

Message persistence

A key attribute of a message is its **persistence**. A message is either persistent or non-persistent. This attribute tells the Queue Manager how important the message is.

Persistent	Non-persistent
	

Queues

The multiple ways of referencing queues builds in application portability – when you want to change a queue that the application uses, you don't have to change the application itself

Only local queues that are defined as a QLOCAL queue type hold messages

Local queue **QLOCAL**	A local queue is a definition of both a queue and the set of messages that are associated with the queue. The queue manager that hosts the queue receives messages in its local queues.
Remote queue **QREMOTE**	Remote queue definitions are definitions on the local queue manager of queues that belong to another queue manager.
Alias queue **QALIAS**	Alias queues are additional definitions of existing queues. You create alias queue definitions that refer to actual local queue, but you can name the alias queue definition differently
Model queue **QMODEL**	A model queue is a template for queues that you want the queue manager to create dynamically as required.

What is a Queue? – More detail

Local queue
QLOCAL

Remote queue
QREMOTE

Alias queue
QALIAS

Model queue
QMODEL

More queues

Transmit or transmission queue

Local queue with its usage attribute set to XMITQ in the queue definition

Dead-letter queue

Local queue that is identified to the queue manager as its dead-letter queue to hold undeliverable messages

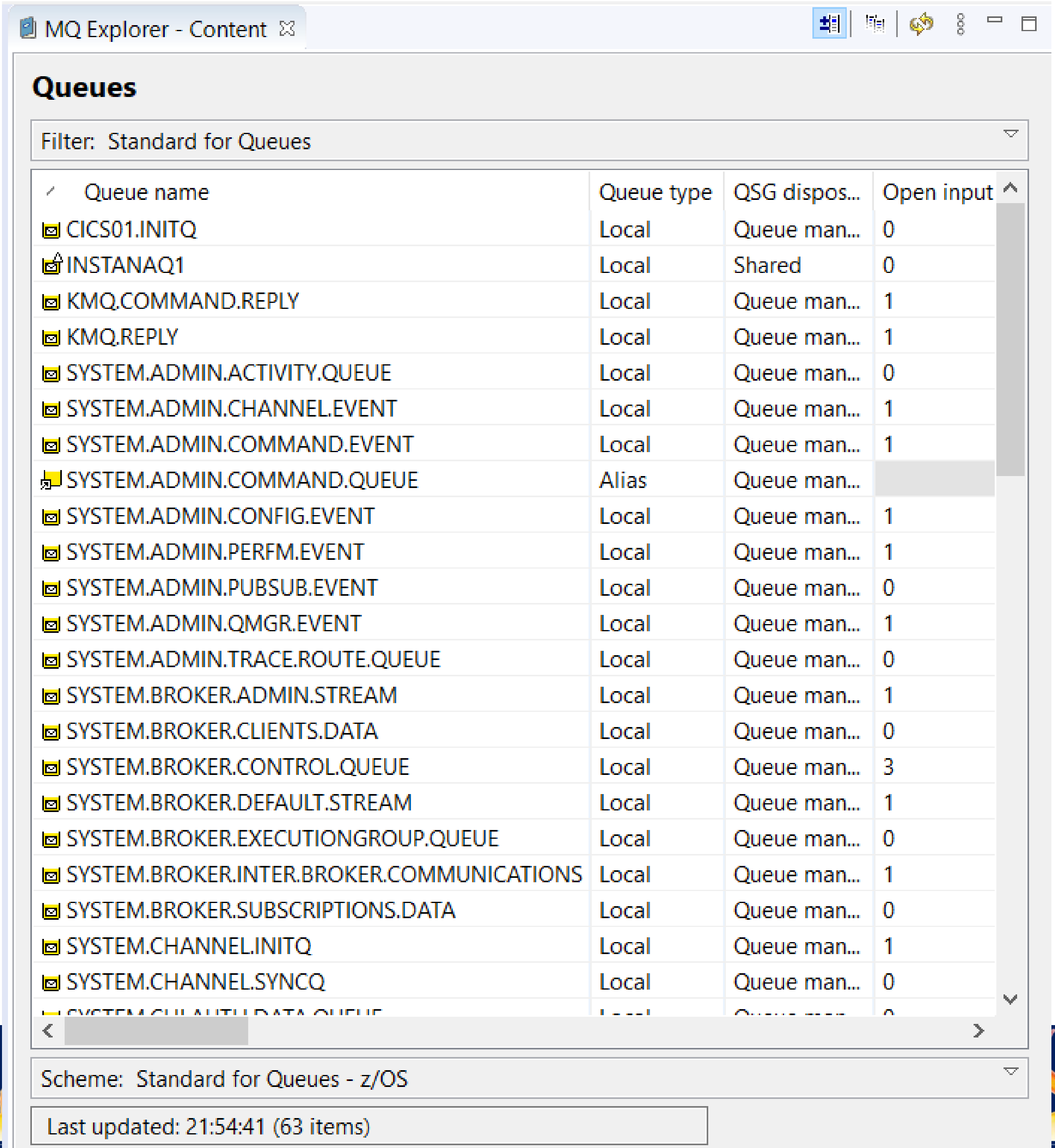
Initiation queues

Identified as an initiation queue in a definition of another local queue

Associated with triggering

Queues starting with “SYSTEM”

Queues that are dedicated to the queue manager for management purposes



MQ Explorer - Content

Queues

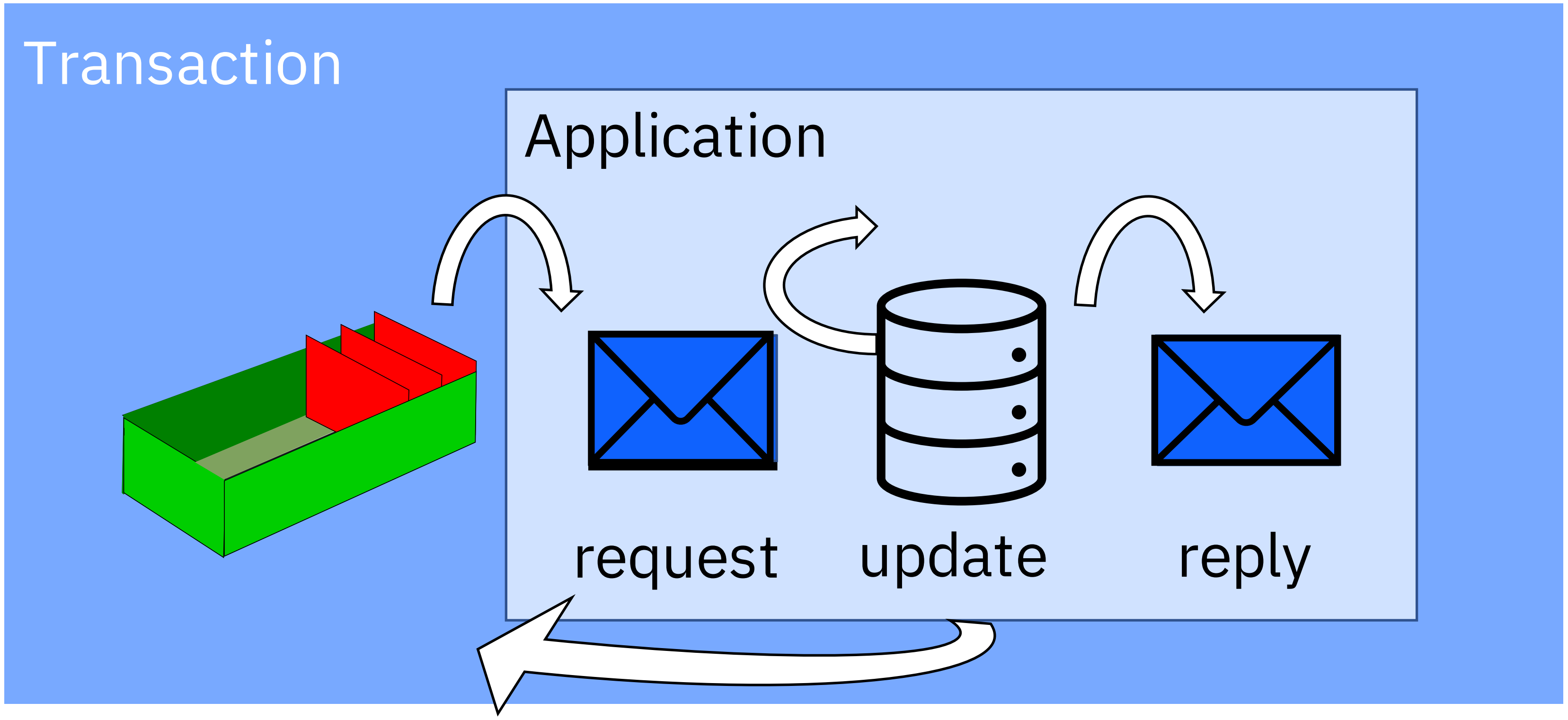
Filter: Standard for Queues

Queue name	Queue type	QSG dispos...	Open input
CICS01.INITQ	Local	Queue man...	0
INSTANAQ1	Local	Shared	0
KMQ.COMMAND.REPLY	Local	Queue man...	1
KMQ.REPLY	Local	Queue man...	1
SYSTEM.ADMIN.ACTIVITY.QUEUE	Local	Queue man...	0
SYSTEM.ADMIN.CHANNEL.EVENT	Local	Queue man...	1
SYSTEM.ADMIN.COMMAND.EVENT	Local	Queue man...	1
SYSTEM.ADMIN.COMMAND.QUEUE	Alias	Queue man...	
SYSTEM.ADMIN.CONFIG.EVENT	Local	Queue man...	1
SYSTEM.ADMIN.PERFM.EVENT	Local	Queue man...	1
SYSTEM.ADMIN.PUBSUB.EVENT	Local	Queue man...	0
SYSTEM.ADMIN.QMGR.EVENT	Local	Queue man...	1
SYSTEM.ADMIN.TRACE.ROUTE.QUEUE	Local	Queue man...	0
SYSTEM.BROKER.ADMIN.STREAM	Local	Queue man...	1
SYSTEM.BROKER.CLIENTS.DATA	Local	Queue man...	0
SYSTEM.BROKER.CONTROL.QUEUE	Local	Queue man...	3
SYSTEM.BROKER.DEFAULT.STREAM	Local	Queue man...	1
SYSTEM.BROKER.EXECUTIONGROUP.QUEUE	Local	Queue man...	0
SYSTEM.BROKER.INTER.BROKER.COMMUNICATIONS	Local	Queue man...	1
SYSTEM.BROKER.SUBSCRIPTIONS.DATA	Local	Queue man...	0
SYSTEM.CHANNEL.INITQ	Local	Queue man...	1
SYSTEM.CHANNEL.SYNCQ	Local	Queue man...	0

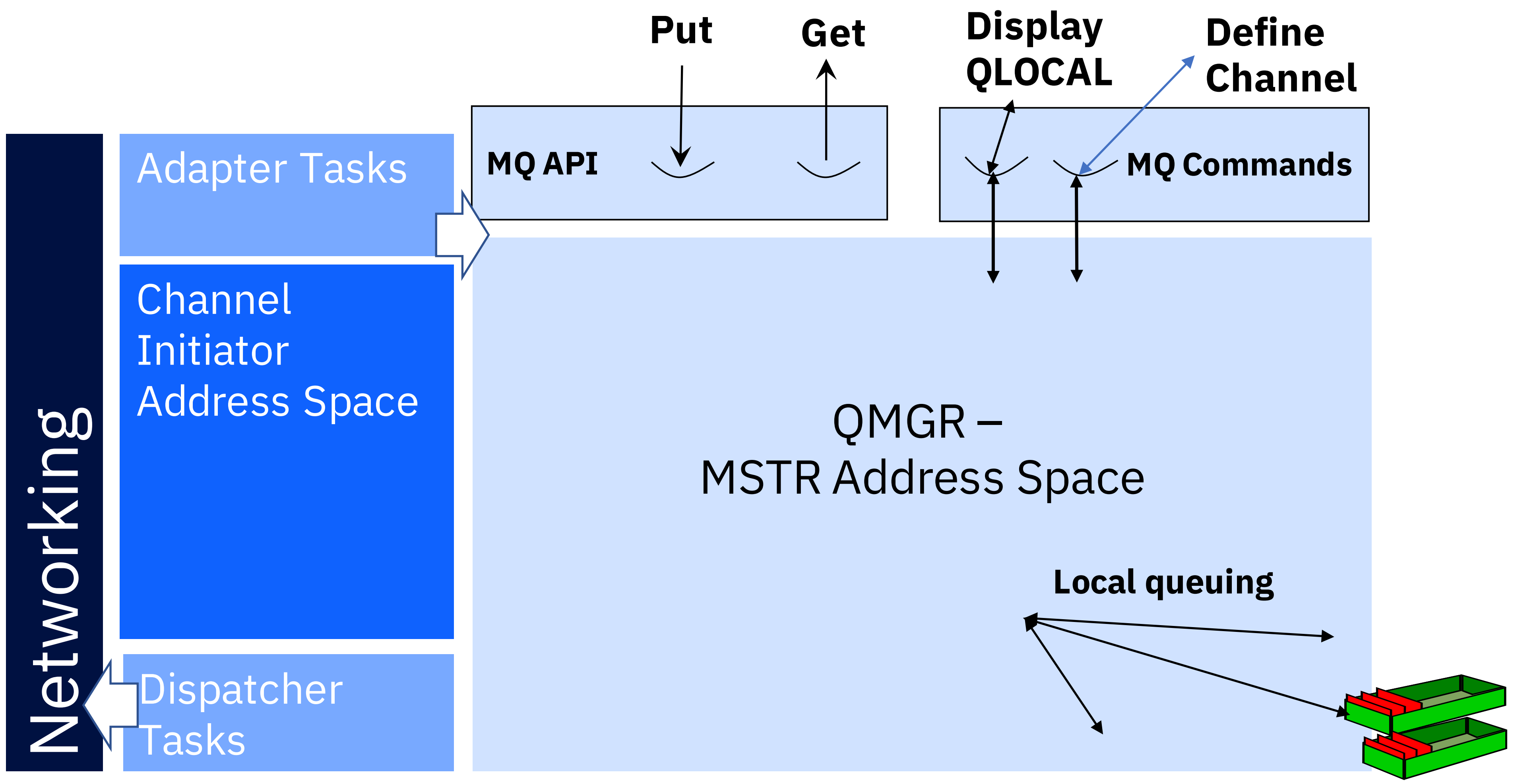
Scheme: Standard for Queues - z/OS

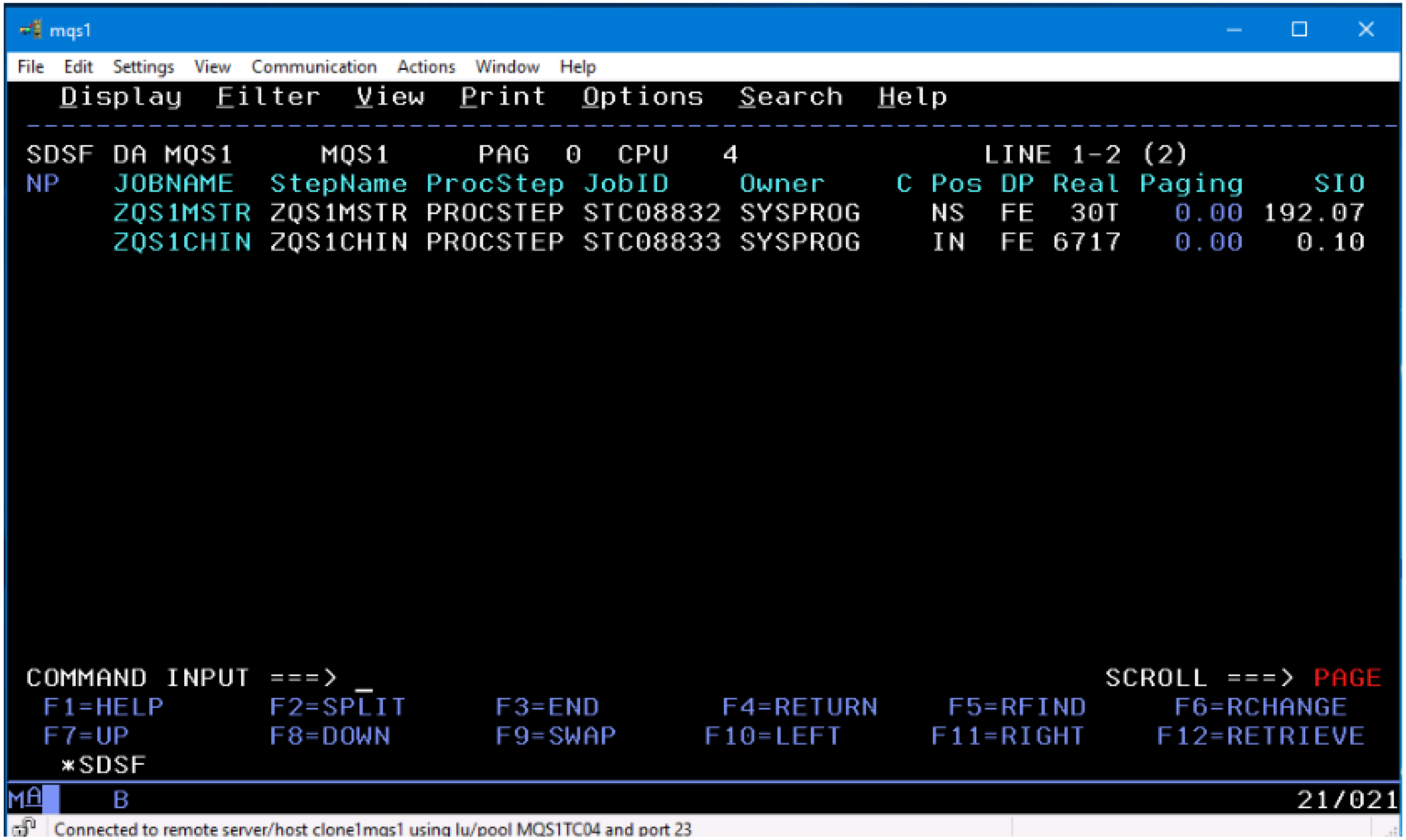
Last updated: 21:54:41 (63 items)

Transaction support in queuing



What is a Queue Manager on z/OS?





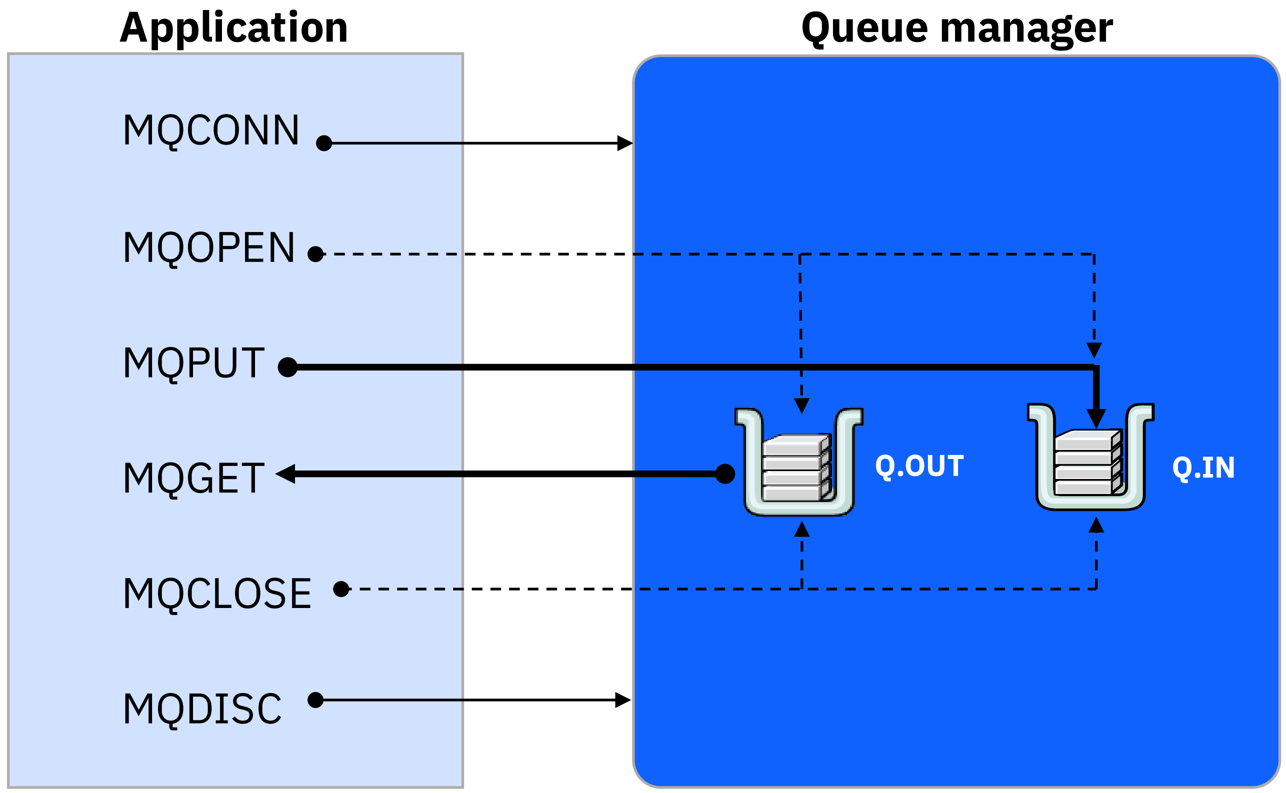
```
  Display  Filter  View  Print  Options  Search  Help
 -------------------------------------------------------------------------------
 SDSF DA MQS1      MQS1      PAG   0   CPU    4              LINE 1-2 (2)
 NP    JOBNAME   StepName ProcStep JobID     Owner     C Pos DP Real  Paging    SIO
       ZQS1MSTR ZQS1MSTR PROCSTEP STC08832 SYSPROG      NS  FE   30T    0.00 192.07
       ZQS1CHIN ZQS1CHIN PROCSTEP STC08833 SYSPROG      IN  FE 6717    0.00   0.10















 COMMAND INPUT ===> _                                          SCROLL ===> PAGE
  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
   *SDSF
MA      B                                                                21/021
```

Connected to remote server/host clone1mqs1 using lu/pool MQS1TC04 and port 23

Data storage under the hood

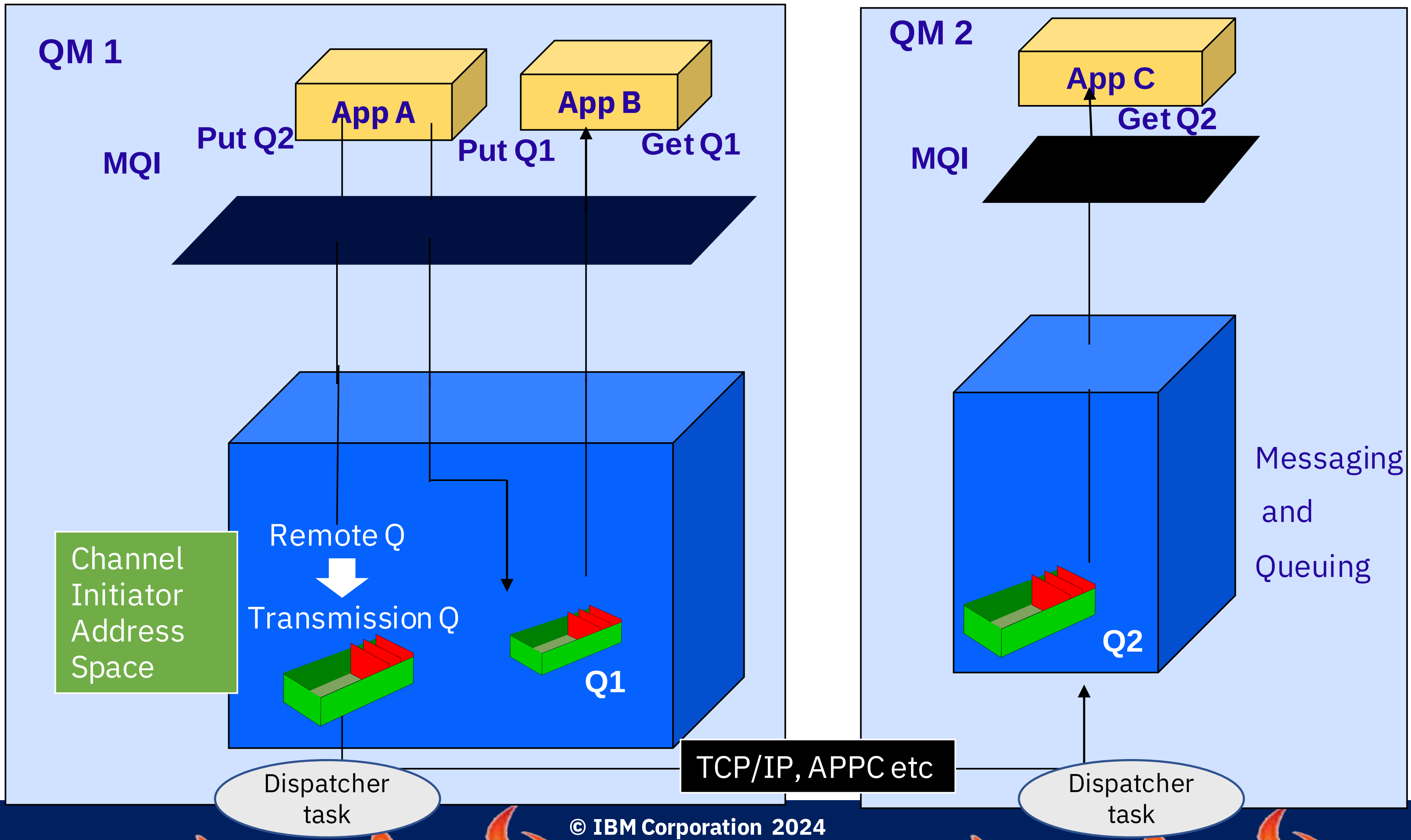
- Buffer pools – temporary data caches for short-lived messages
- Page sets – VSAM data sets that store messages for a queue manager
  - Store messages and object definitions

MQ API commands

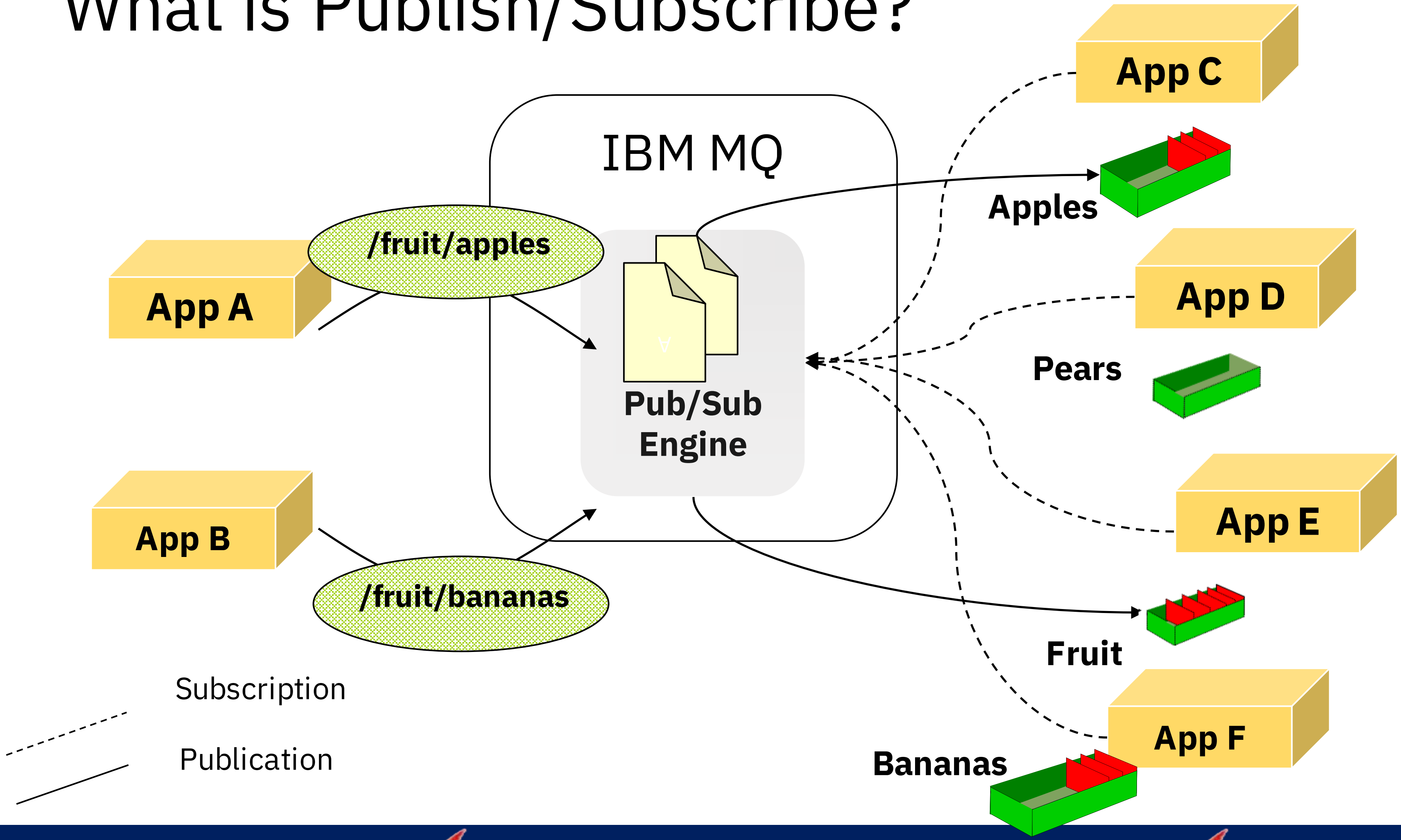


IBM MQ provides numerous sample programs

MQ Channels

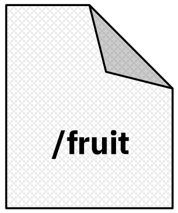

QM 1
App A
App B
Put Q2
Put Q1
Get Q1
MQI
Remote Q
Transmission Q
Q1
Channel Initiator Address Space
Dispatcher task
TCP/IP, APPC etc
QM 2
App C
Get Q2
MQI
Q2
Messaging and Queuing
Dispatcher task

What is Publish/Subscribe?



What are Topic Strings and Topic Objects?

- Topic Object
  - Is a predefined MQ object with a 48-character name
  - Allows you to assign specific non default information for the pub/sub environment
  - Has a topic string as an attribute
  - Is a security control point



- Topic String
  - Is a character string
  - Can be made up of any characters
  - Is case sensitive
    - /fruit/apples
  - Is the ‘subject matter’ for Publications and Subscriptions

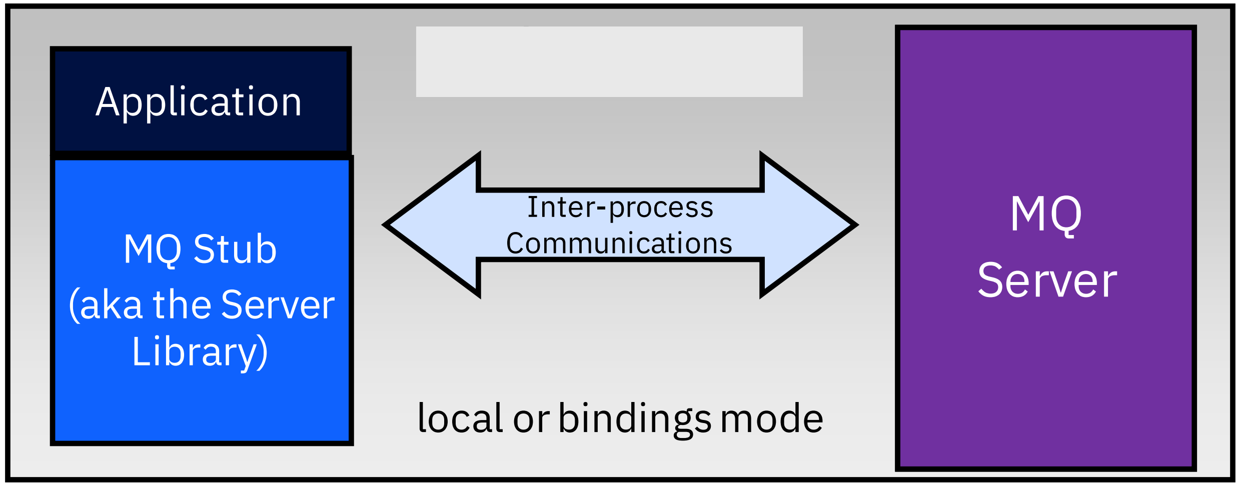
Concept check

- What are the two parts of a message?
  - Tag and content
  - Meat and potatoes
  - MQMD and payload
  - Metadata and data
- What is a remote queue?
  - A queue associated with physical storage
  - A queue defined to another queue manager
  - A queue name that resolves to another queue
- What is a dead-letter queue?

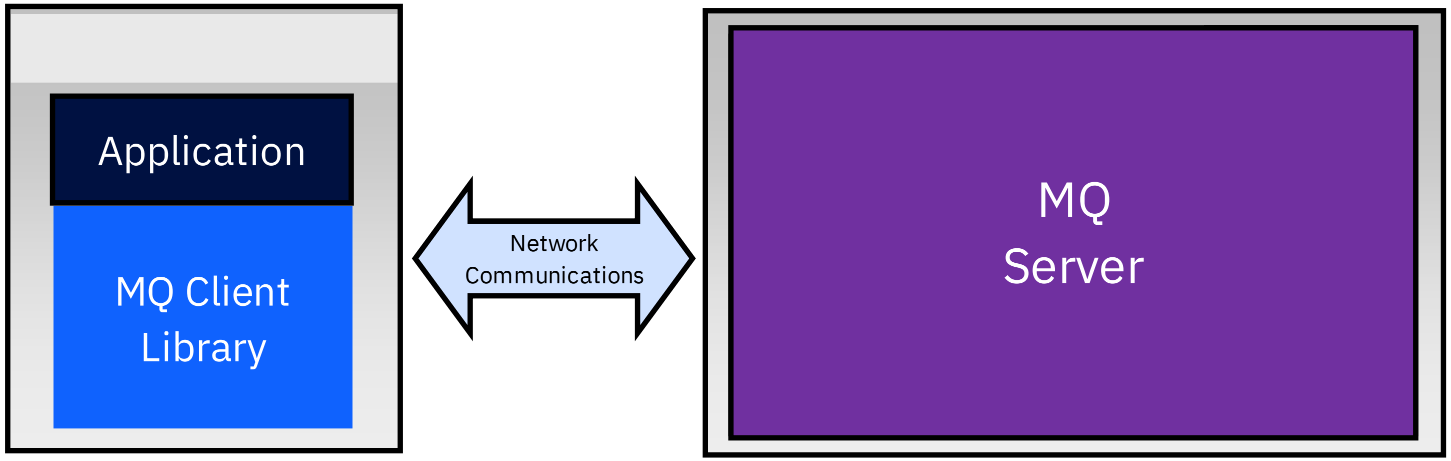
How are queue managers arranged on z/OS?

What is the Client implementation on z/OS?

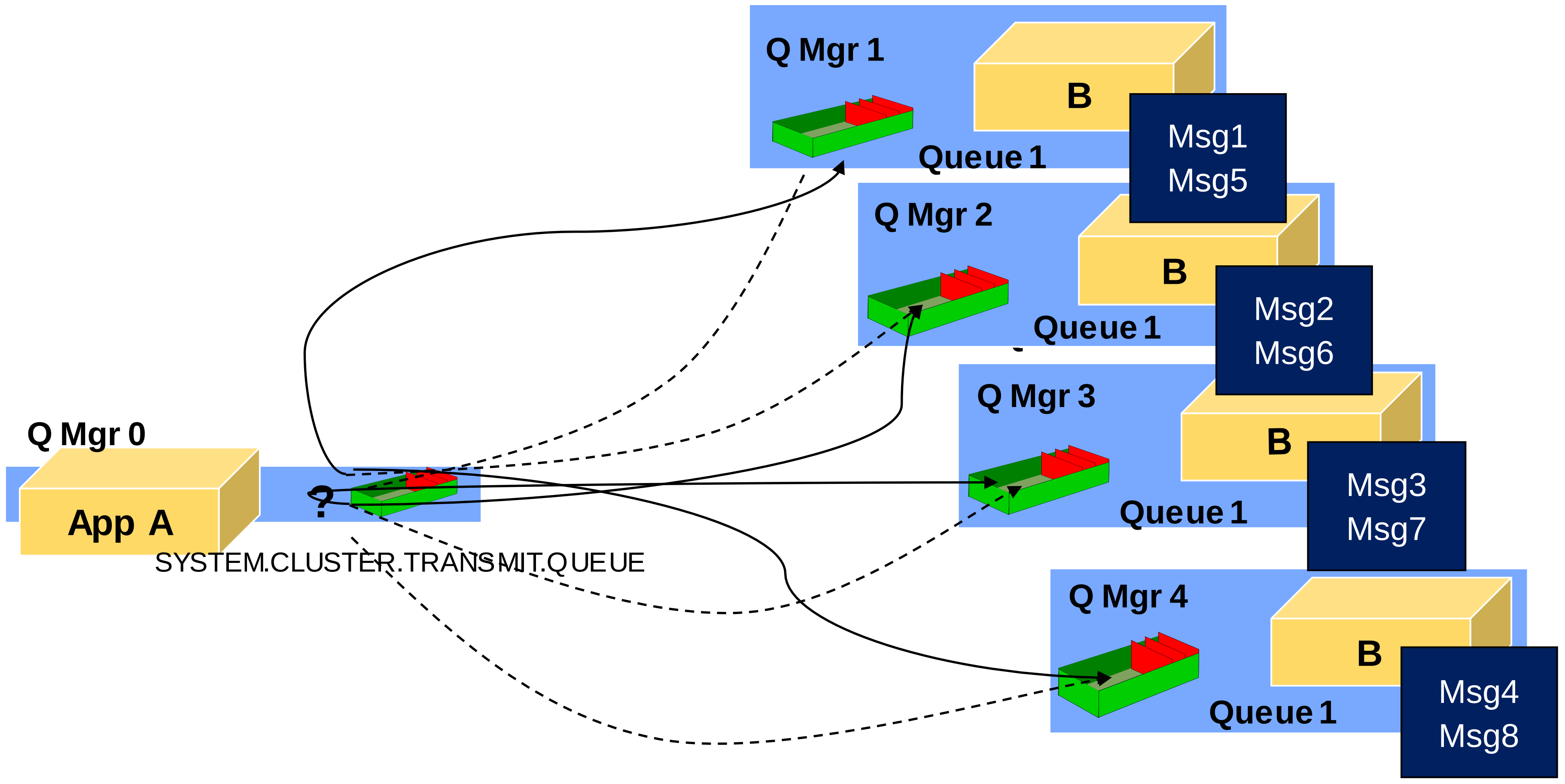
Server Model

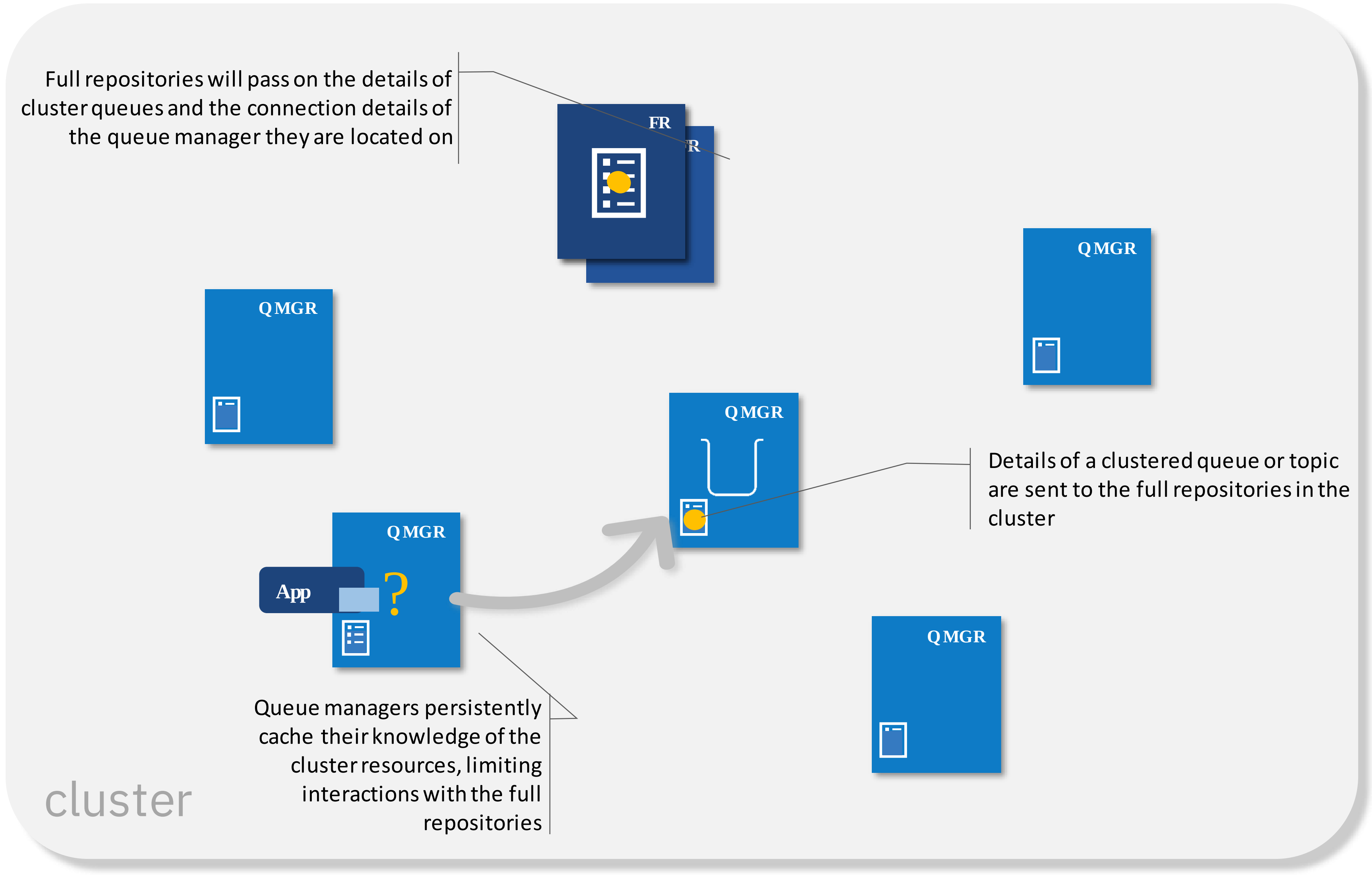


Client Model



How does a cluster normally work?





Full repositories will pass on the details of cluster queues and the connection details of the queue manager they are located on

Details of a clustered queue or topic are sent to the full repositories in the cluster

Queue managers persistently cache their knowledge of the cluster resources, limiting interactions with the full repositories

What is a Queue Manager Cluster?

A **cluster** is a group of queue managers set up in such a way that the queue managers can communicate directly with one another over a single network, without the need for multiple transmission queue, channel, and remote queue definitions.

Each queue manager in the cluster has one or more cluster transmissions queue from which it can transmit messages to other queue managers in the cluster.

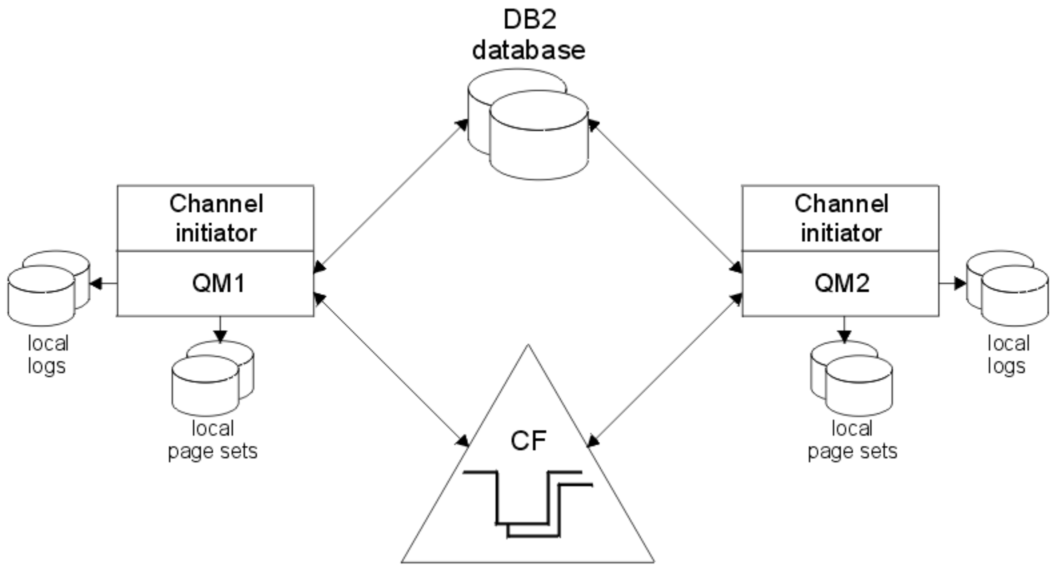
Queue managers in a cluster can be at different versions of MQ (as long as that version does support clustering) and on different platforms.

A cluster is composed of:

- Two full repository queue managers
- Cluster sender and receiver channels
- Partial repository queue managers
- Cluster defined objects (queues, topics)

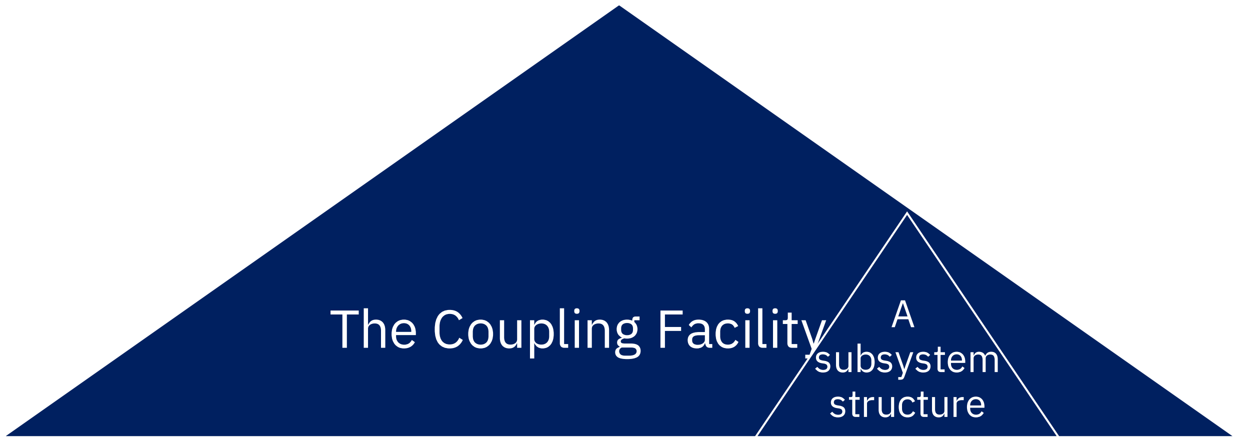
Shared queue terms

A unique feature to MQ on z/OS, shared queues were designed and built to provide **continuous availability** for MQ messages.



Coupling Facility

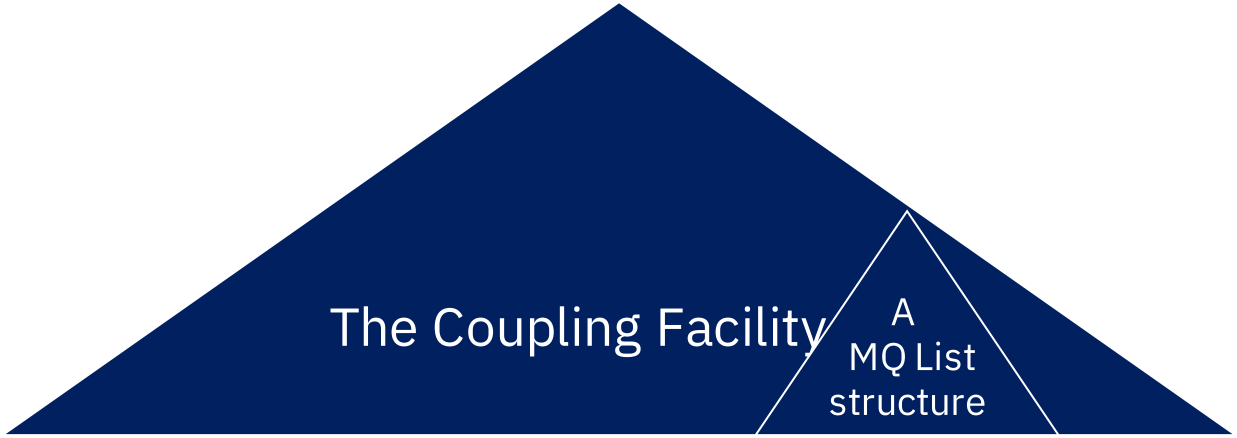
A **coupling facility** is special hardware and software that allow multiple systems to access the same data. It is unique to z/OS, and is required for a parallel sysplex environment.



List Structure

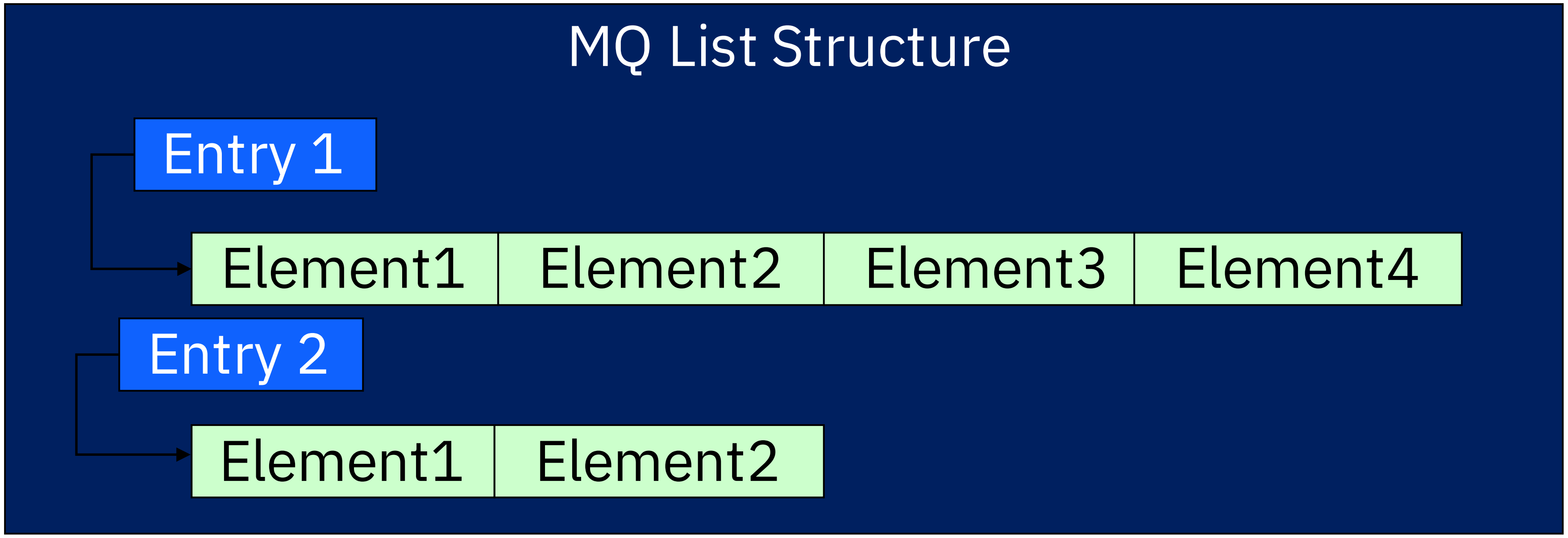
A **list structure** is a data holding structure in the Coupling Facility used by MQ, IMS and DB2 to hold ‘lists’ of data. For MQ, a single list structure can host up to 512 queues.

Messages are held on the list structures

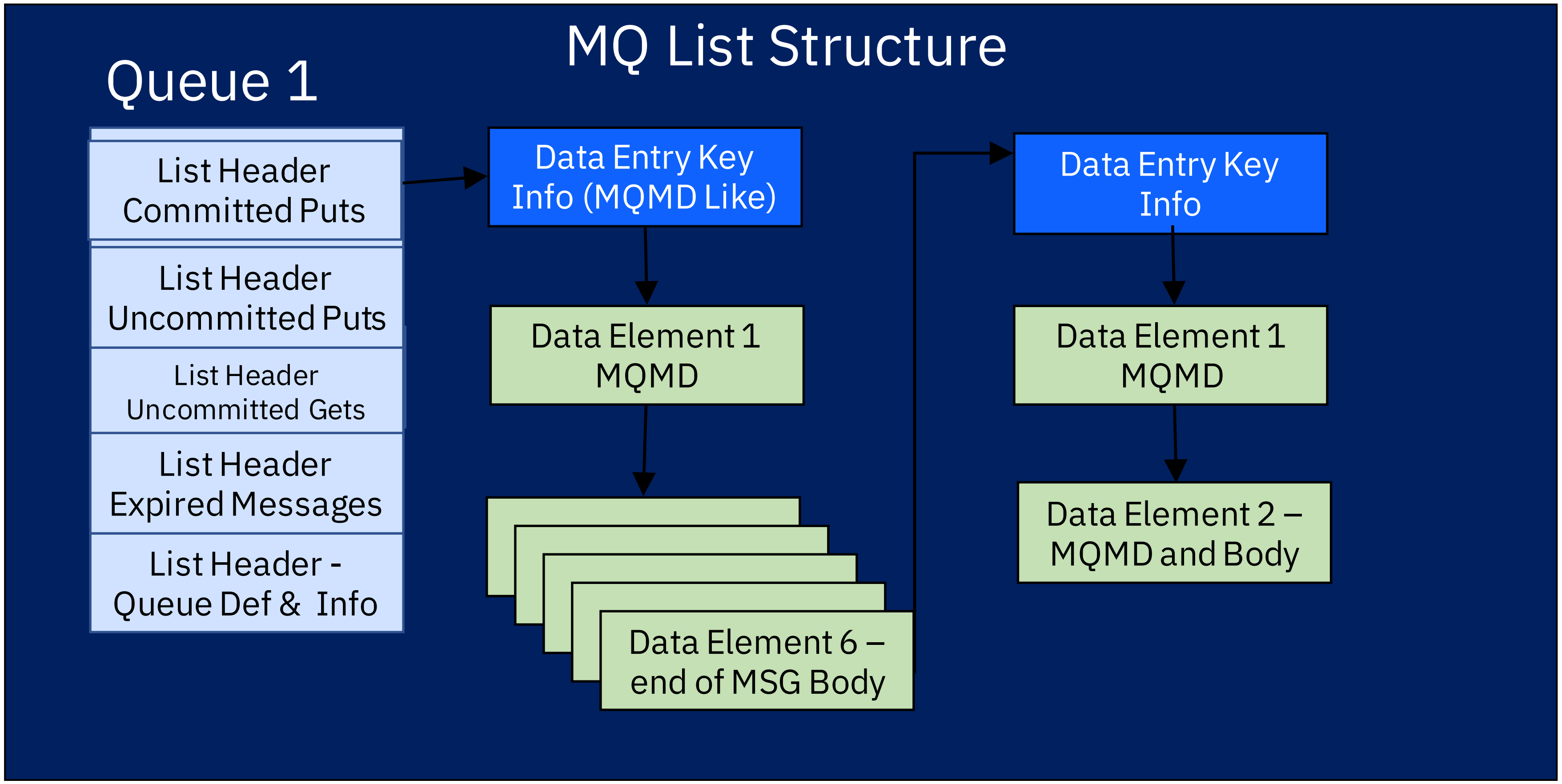


Elements and Entries

- An **entry** is the anchor of an individual message in the list structure. It is 256-bytes and is mostly pointers to the elements.
- The **elements** are the chunks of the message in the list structure.



At a deeper level


MQ List Structure
Queue 1
List Header Committed Puts
List Header Uncommitted Puts
List Header Uncommitted Gets
List Header Expired Messages
List Header - Queue Def & Info
Data Entry Key Info (MQMD Like)
Data Element 1 MQMD
Data Element 6 – end of MSG Body
Data Entry Key Info
Data Element 1 MQMD
Data Element 2 – MQMD and Body

MQ Queue Sharing terms

A **Queue Sharing Group** is a logical association of queue managers in a Sysplex. These queue managers are connected to MQ list structures and a DB2 Data sharing group. This allows them to share queues and their messages, to treat any queue defined on the CF as if it is local (can do both MQGETs and MQPUTs).

- There can be up to 32 queue managers in a QSG.

A **shared queue** is a queue defined on a Coupling Facility structure

- Available to every queue manager on the queue shared group as if it is a local queue.

CFSTRUCT is a MQ object that defines the Coupling Facility list structure to MQ. Queues are defined to the list structure.

```
DEFINE QLOCAL(queue-name) QSGDISP(SHARED) CFSTRUCT(list-str-
name)
```

Compare to: Private Queuing

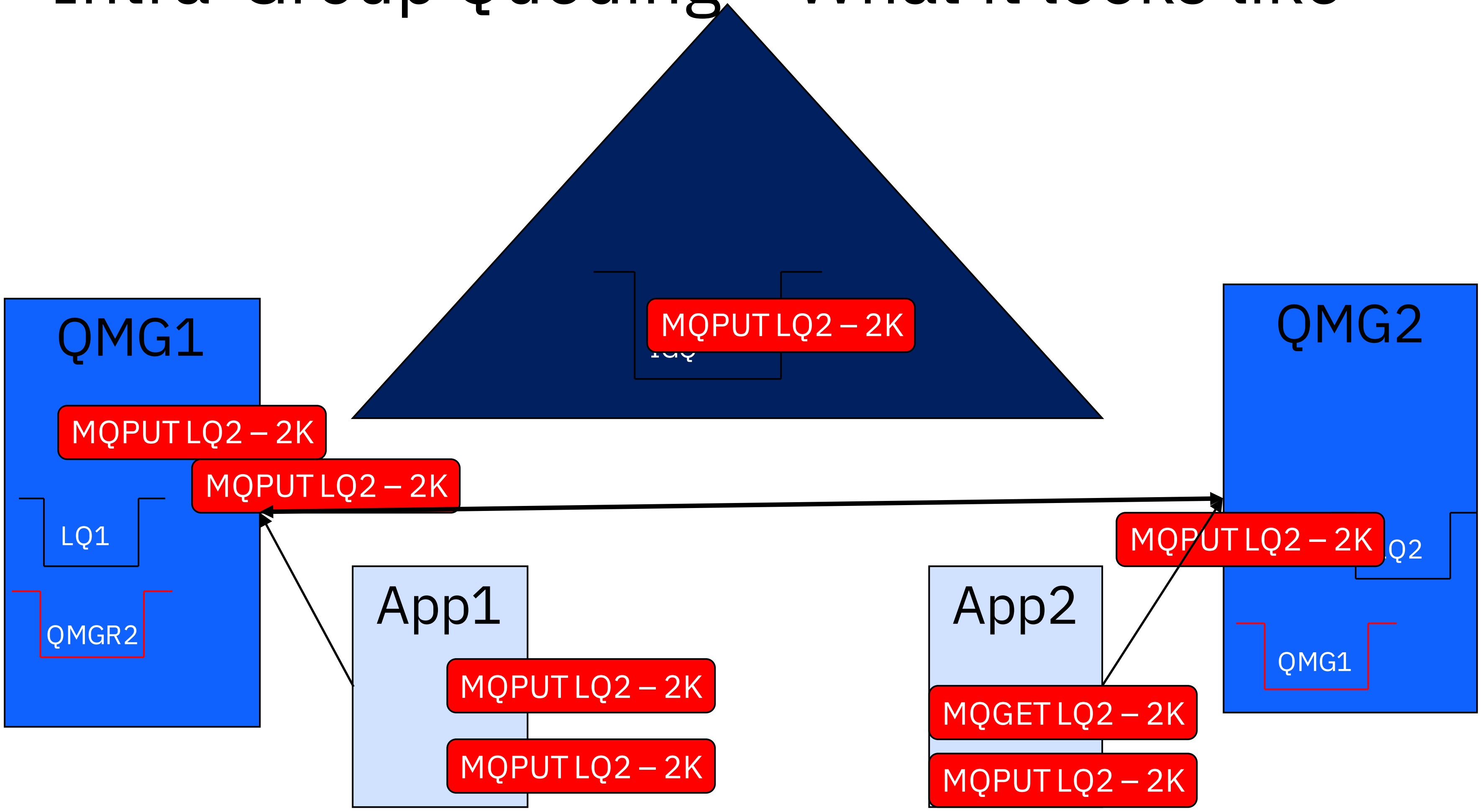
A **private queue** is a local queue defined to and managed by a specific queue manager.

- On z/OS, they use local buffer pools and page sets for their physical message storage.
- On distributed, they use file systems

Messages on private queues are **only available for MQGETs to applications connected to the queue manager where they are defined.**

All local queues are private on distributed queue managers

Intra-Group Queuing – What it looks like



Intra-Group Queuing

IGQ uses the CF to pass messages between queue managers within the same Queue Sharing Group

- Can be more efficient than normal channels
  - Especially for small messages
  - Avoid multi-hopping in most configurations
- Uses the SYSTEM.IGQ.TRANSMIT.QUEUE
- Remote queue and channel definitions are still necessary
- Message size determines whether a message is sent via IGQ or a channel. Message size is controlled on SYSTEM.IGQ.TRANSMIT.QUEUE definition:
  - If the CFSTRUCT used is level 3, the max message size is 63K
  - MAXMSGL can also be adjusted down from the default

Concept check

What is a queue-sharing group?

A) Two or more queue managers sharing message data via list structures

B) A configuration used to influence message distribution across queue managers

C) A configuration used to reduce complexity across MQ on z/OS

What does a dispatcher task do?

A) Helps with the configuration of QSG via list structures

B) Acts as a worker node for channel requests

C) Provides storage for the queue manager

Thank you!

Circular Logging versus Linear Logging

Circular Logging

- Keeps all restart data in a ring of log files
- Using the log to roll back transactions that were in progress

Linear Logging

- Linear logging keeps the log data in a continuous sequence of log files.