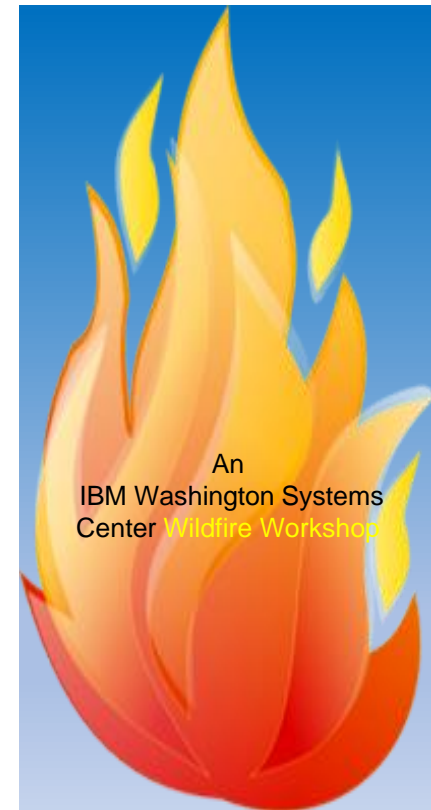




MQPERF1

Intro

Lyn Elkins –
elkinsc@us.ibm.com



Legal Disclaimer

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.



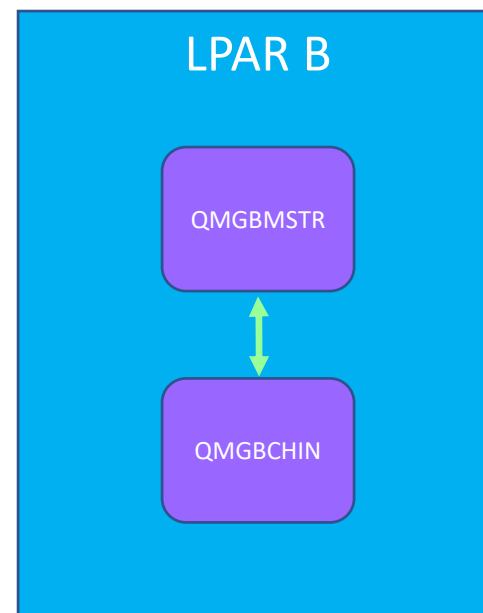
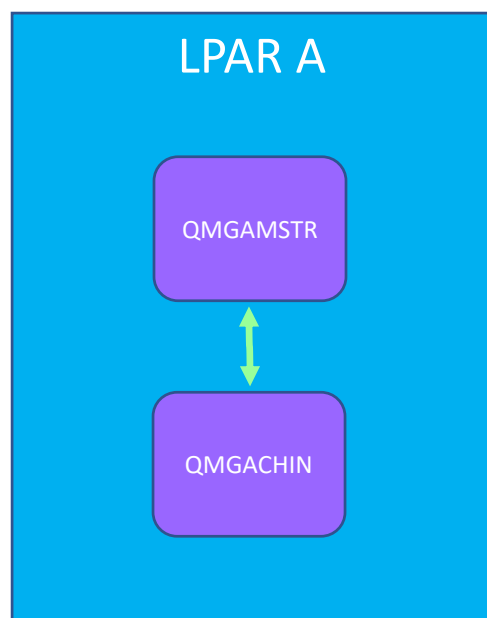
YOUR MILEAGE WILL VARY



A bit of background

To understand performance tuning, and in some measure problems determination – a bit about the internals of the queue manager is helpful. So we are going to start there!

The MQ Topology – What does a z/OS queue manager 'look' like?





A z/OS queue manager

A z/OS queue manager has two address spaces, the queue manager and the channel initiator. They communicate via cross memory services.

Each address space has multiple tasks, that coordinate the requests.

Agenda – QMGR Internals Overview

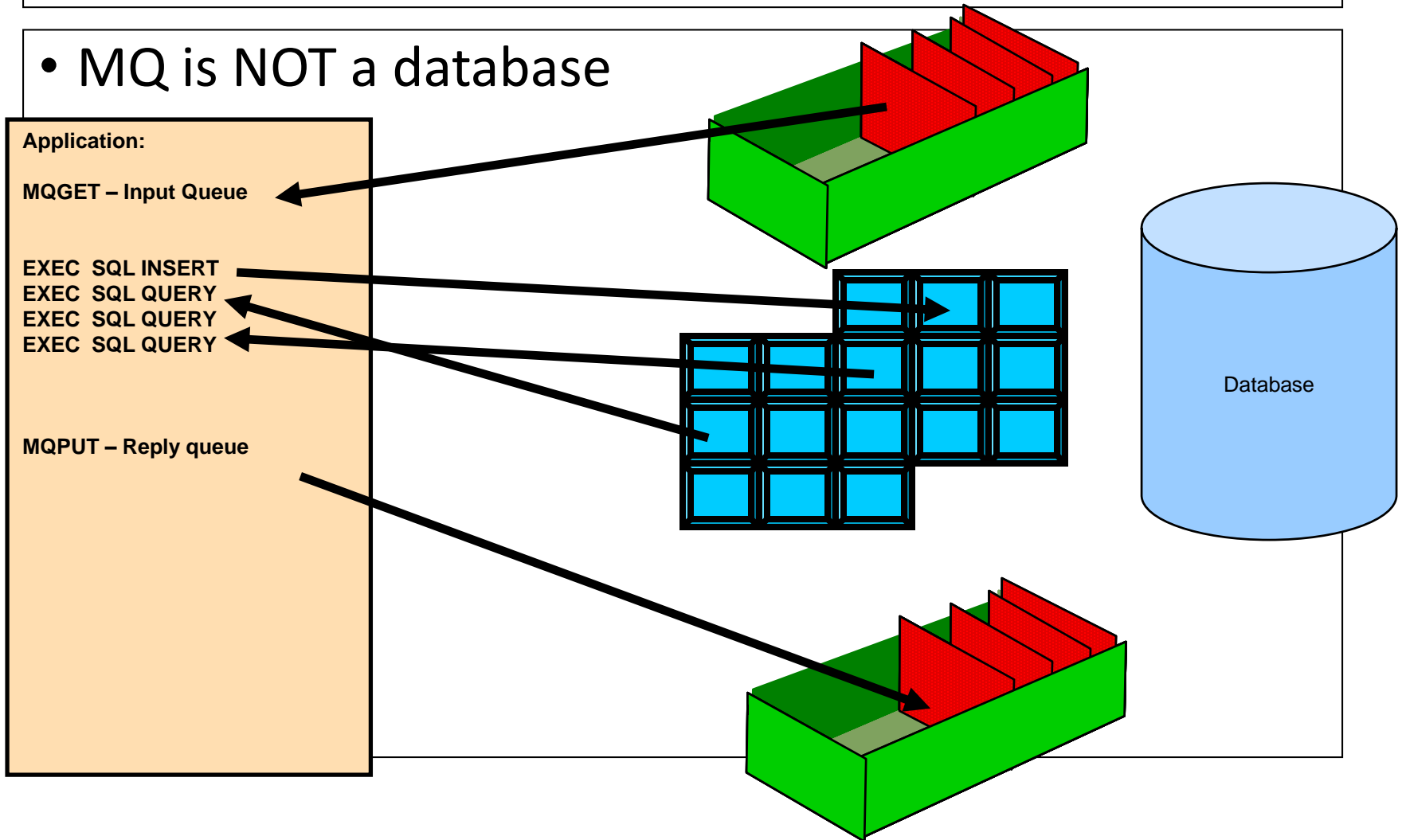
- Why is this important to me?
- One of these things is not like the other
- How are messages stored?
 - Private Queues
 - Shared Queues
- First line managers - the components of a z/OS queue manager
- What happens on a API call?
- Summary

Why is this topic important to me?

- Queue manager performance
 - Knowing how the pieces fit together
- Application performance
 - If the queue manager is not tuned, responsiveness can be affected
- Problem resolution

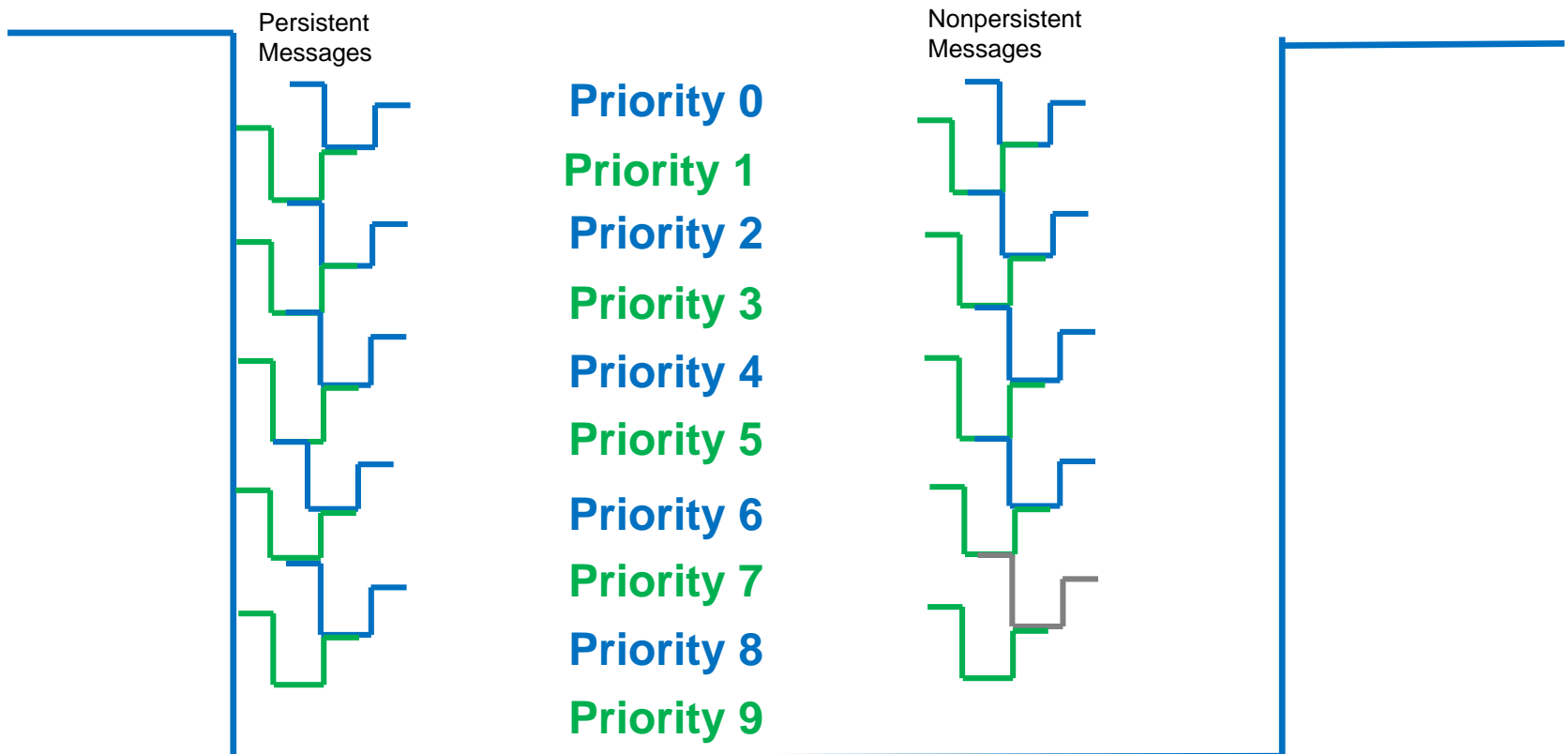
One of these things is not like the other

- MQ is NOT a database

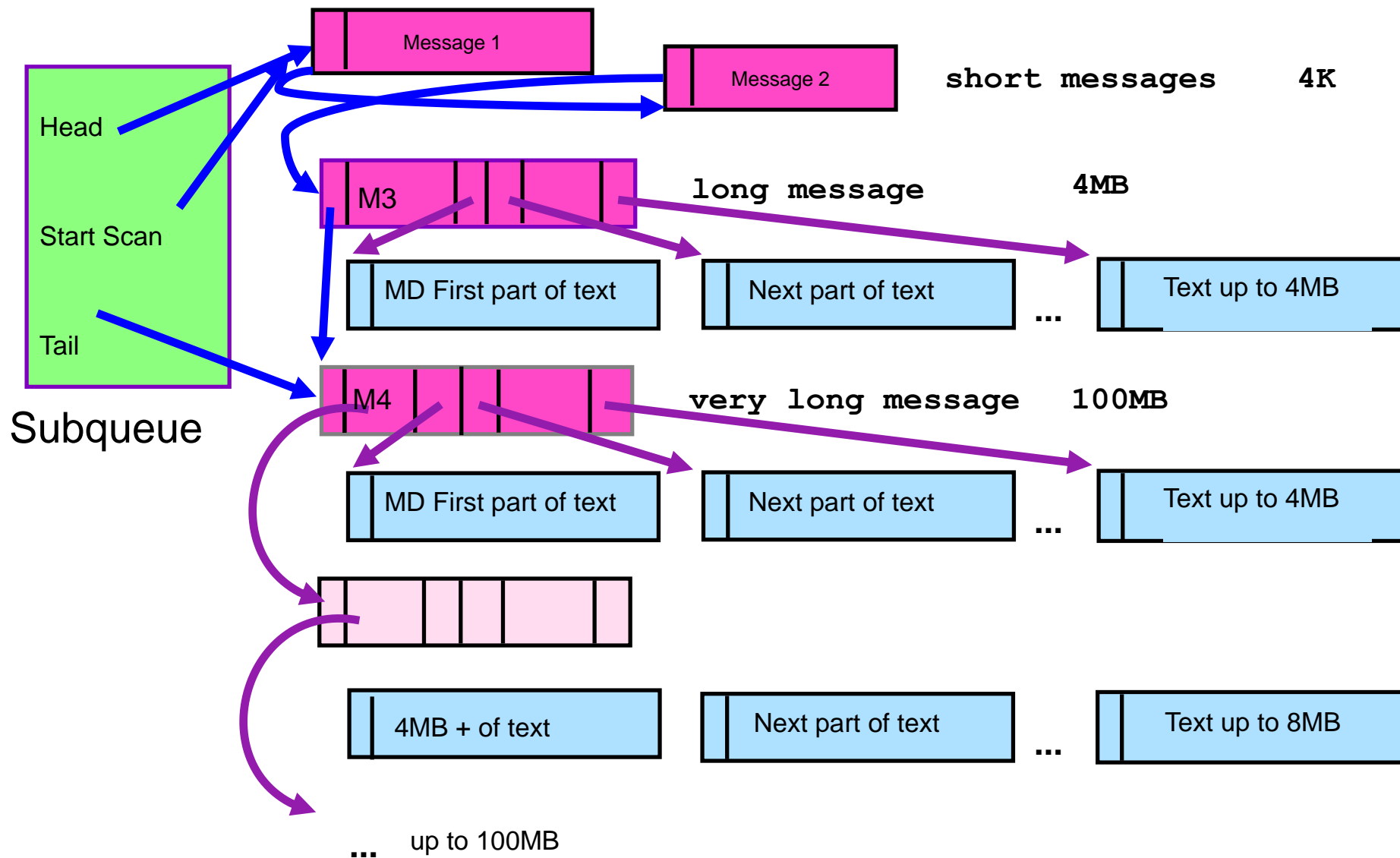


The Internal Representation of a Queue

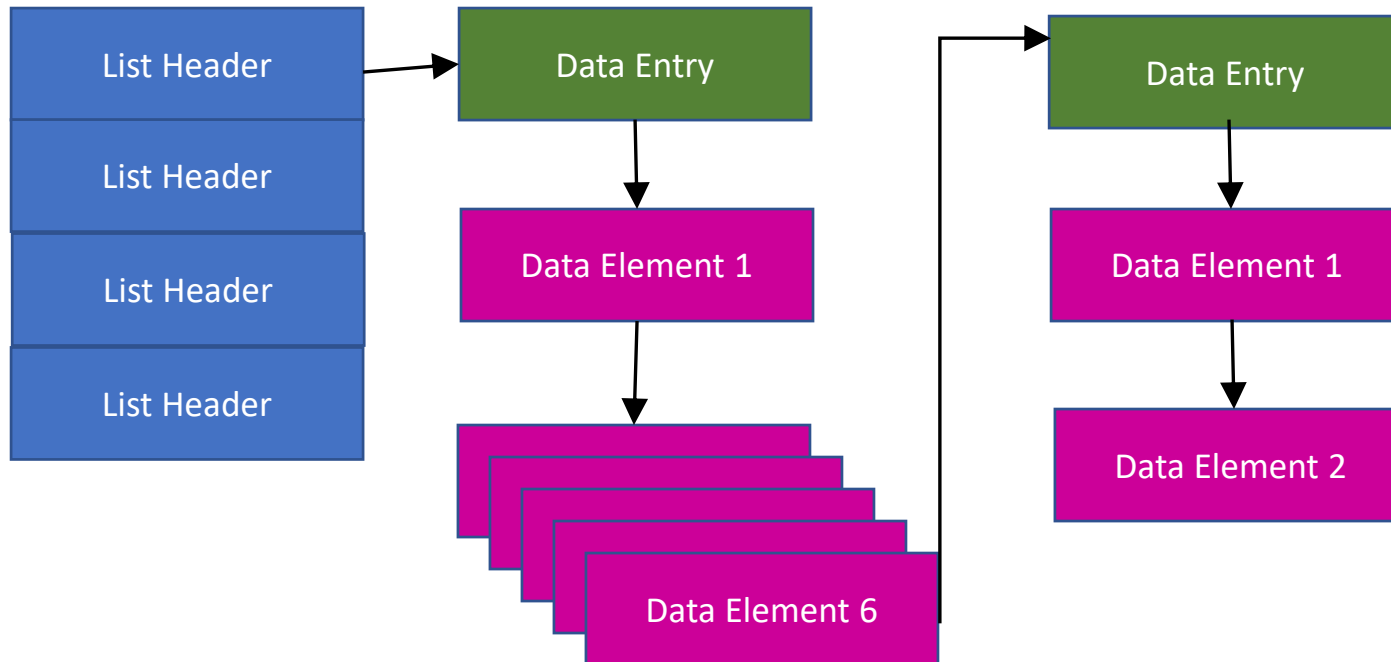
- Sub-queues within a queue



Private Queue Message Storage

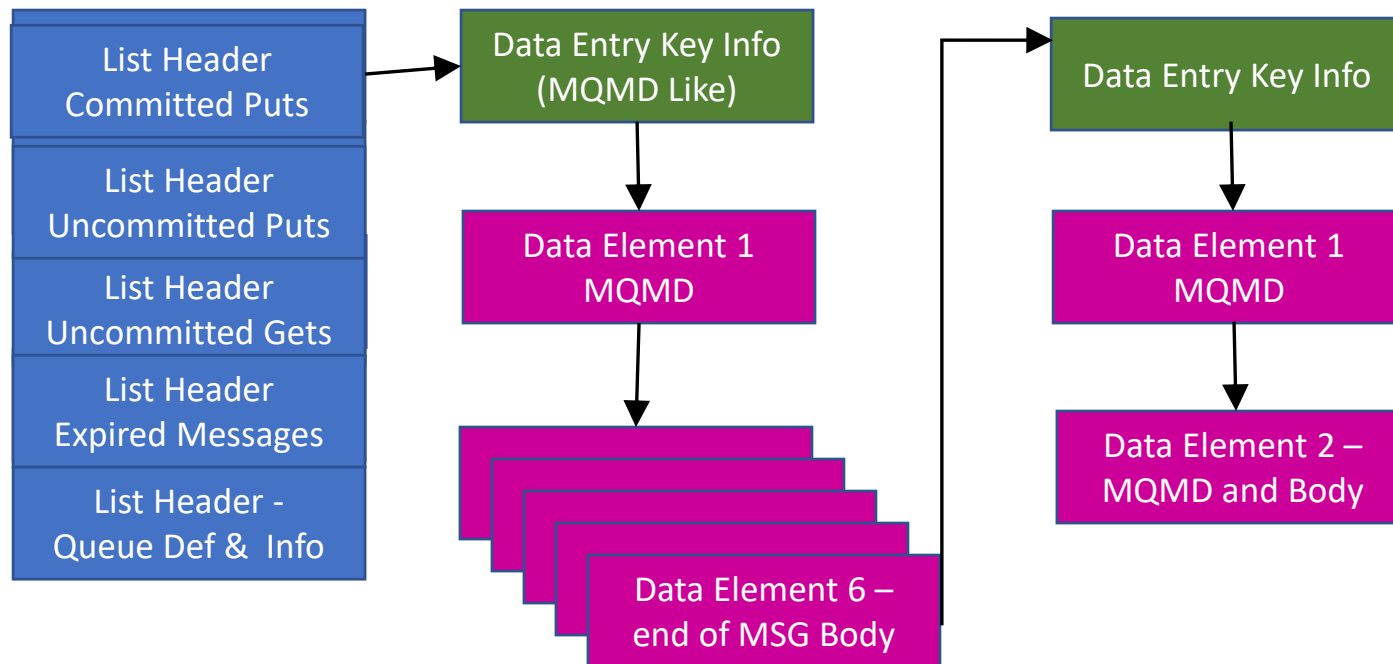


Coupling Facility List Structure



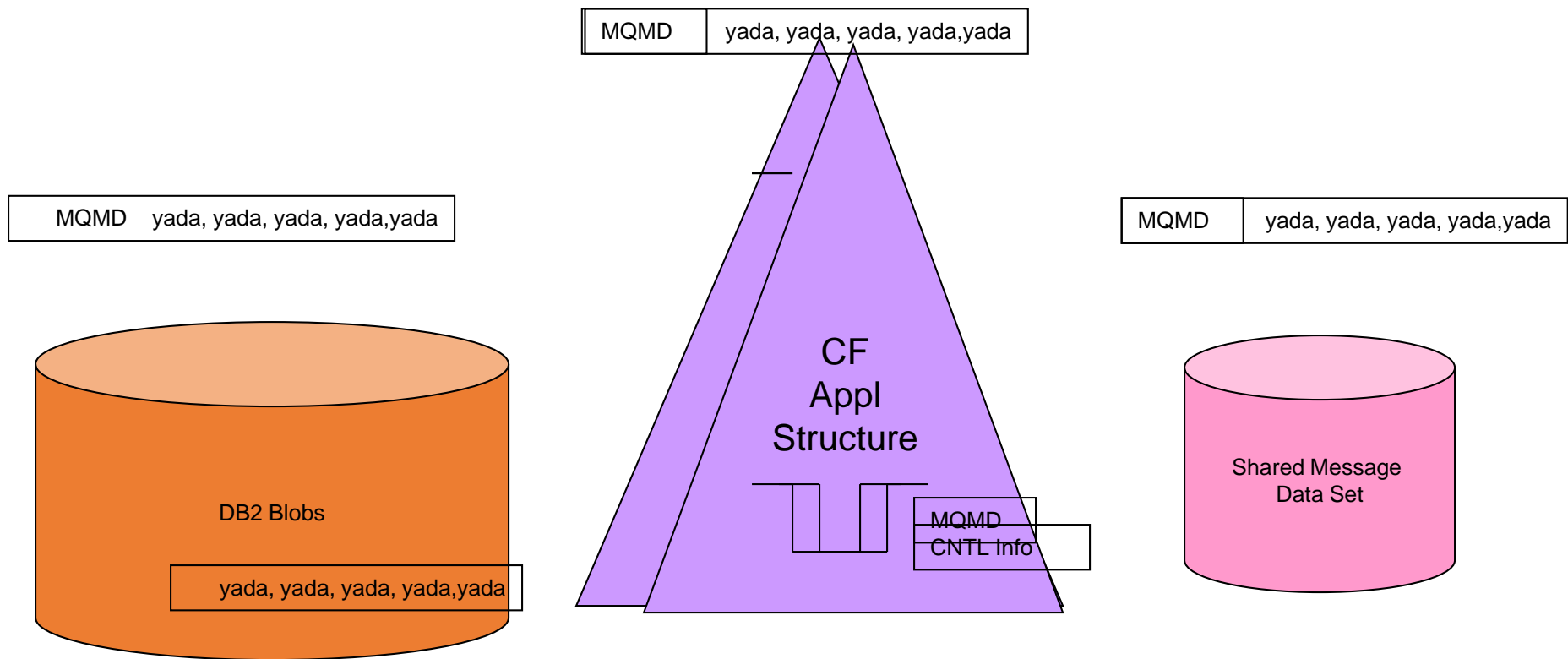
Coupling Facility List Structure

What does a queue look like?



Shared Queue Message Storage

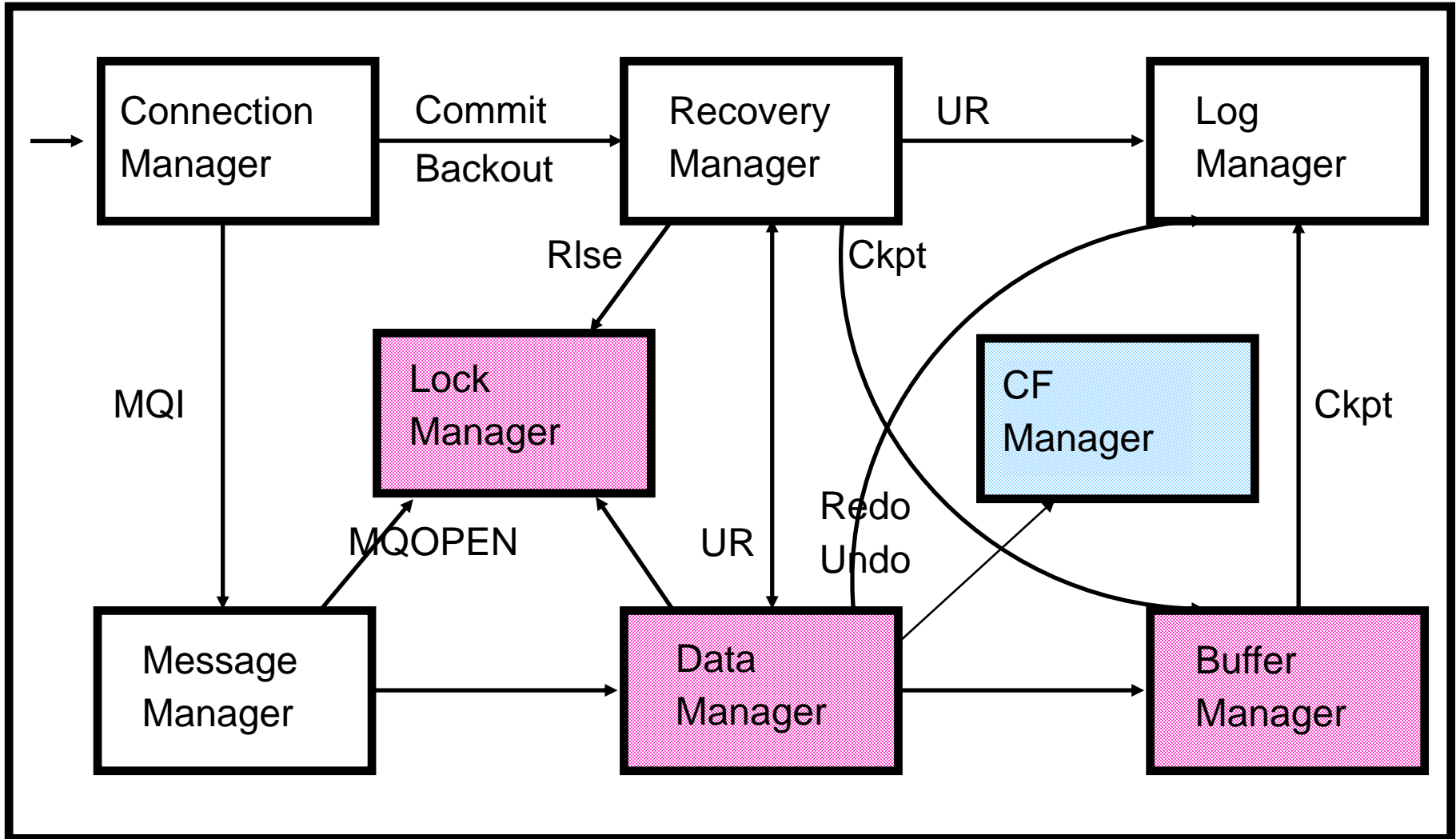
- Messages are stored in one of several ways:
 - Entirely within the list structure
 - Control information (CI) on list structure, message body in DB2
 - CI on list structure, message on Shared Message Data Sets
 - CI and/or message moved to Flash Memory (not shown)



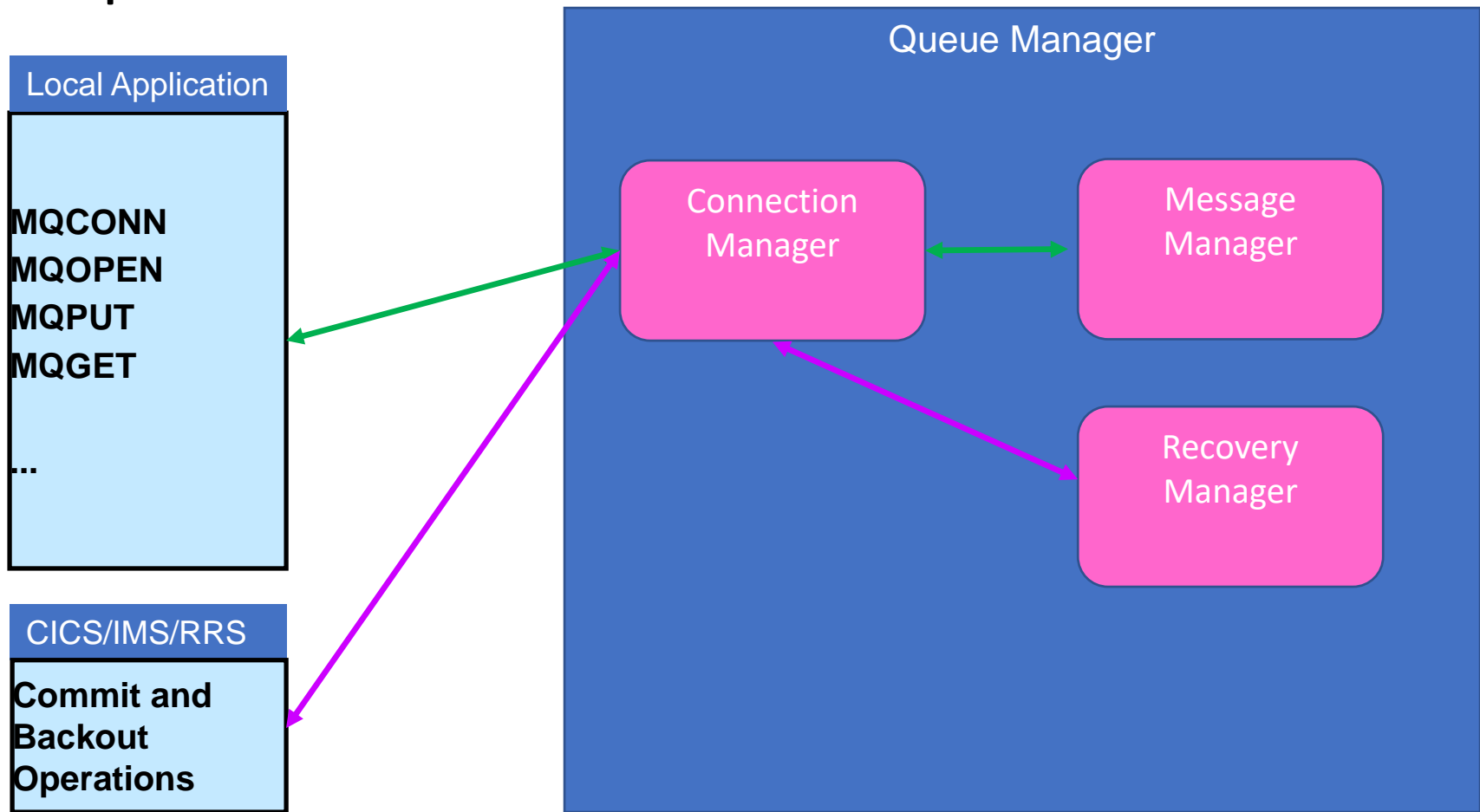
First Line Managers – who does the real work

- To provide the qualities of service that are the basis for WMQ, the real work within the queue manager is divided into logical 'workers' or managers. They interact with the applications and the underlying z/OS resource managers.
- They include:
 - Connection Manager – not the Channel Initiator, but local connections
 - Recovery Manager
 - Log Manager
 - Message Manager
 - Topic manager
 - Data Manager
 - Buffer Manager
 - Lock Manager
 - Storage Manager
 - CF Manager
 - Security Manager.....

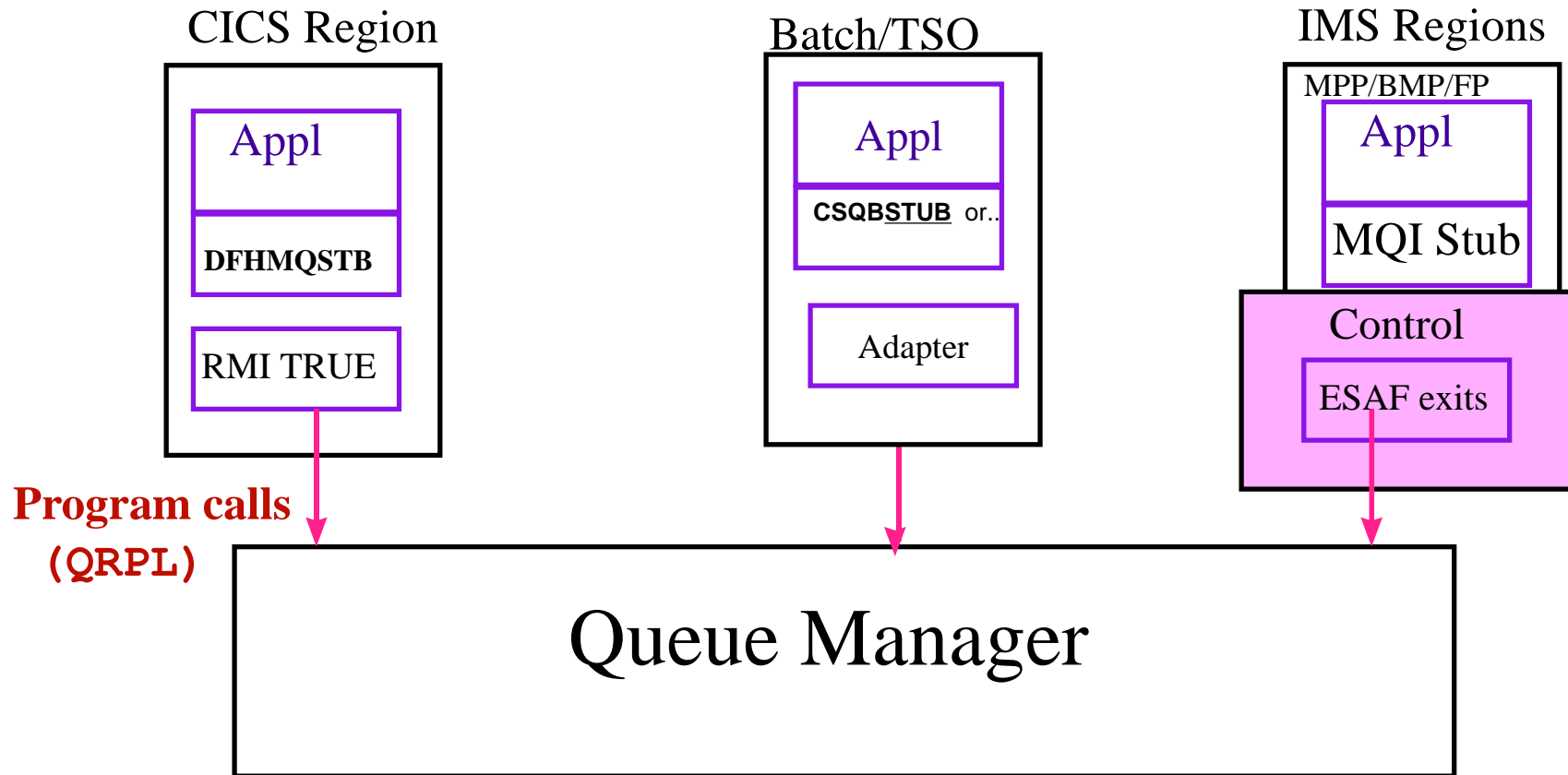
Building Blocks - Resource Managers



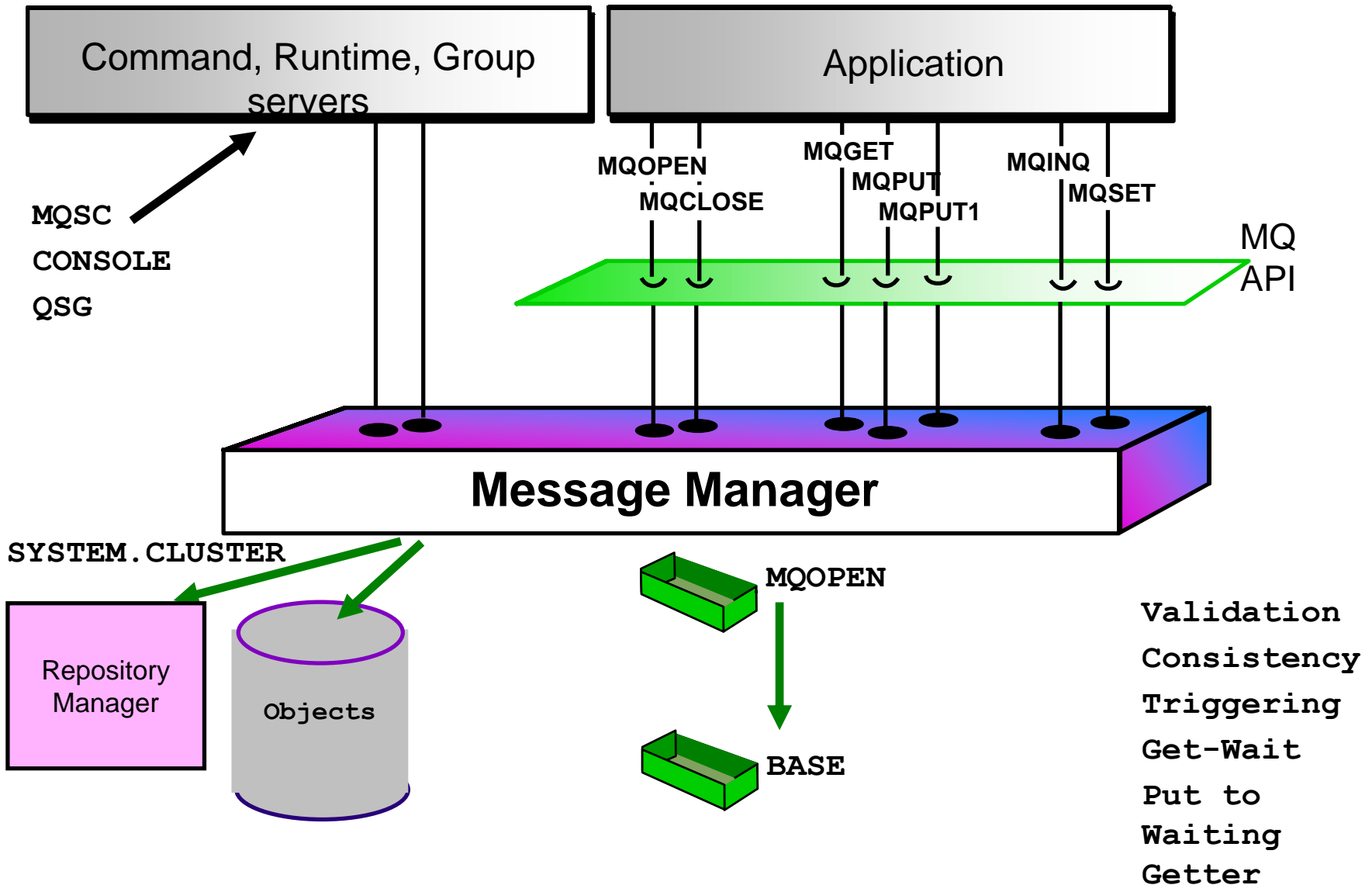
Connection Manager – the entry point



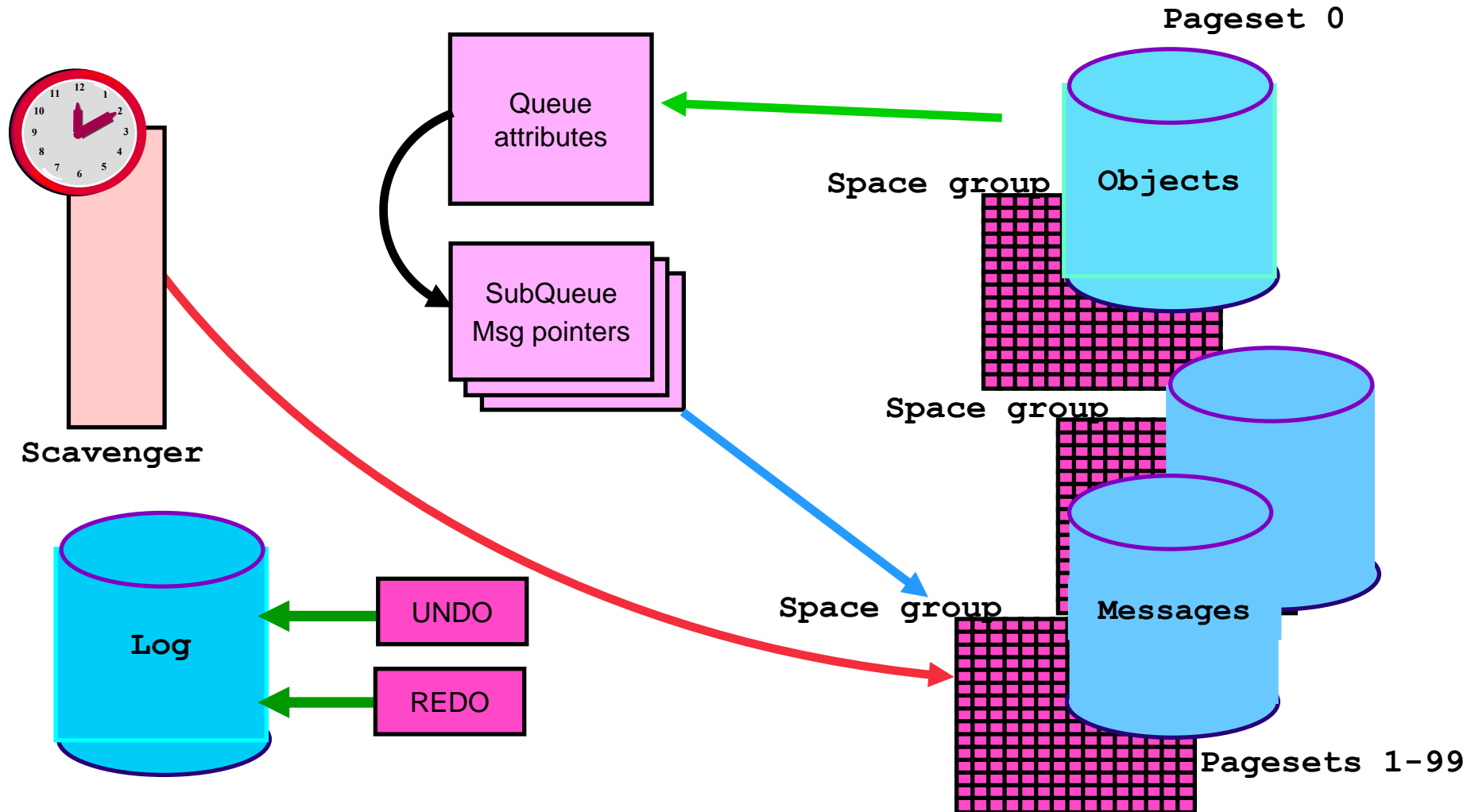
Getting requests into WMQ - Stubs and Adapters



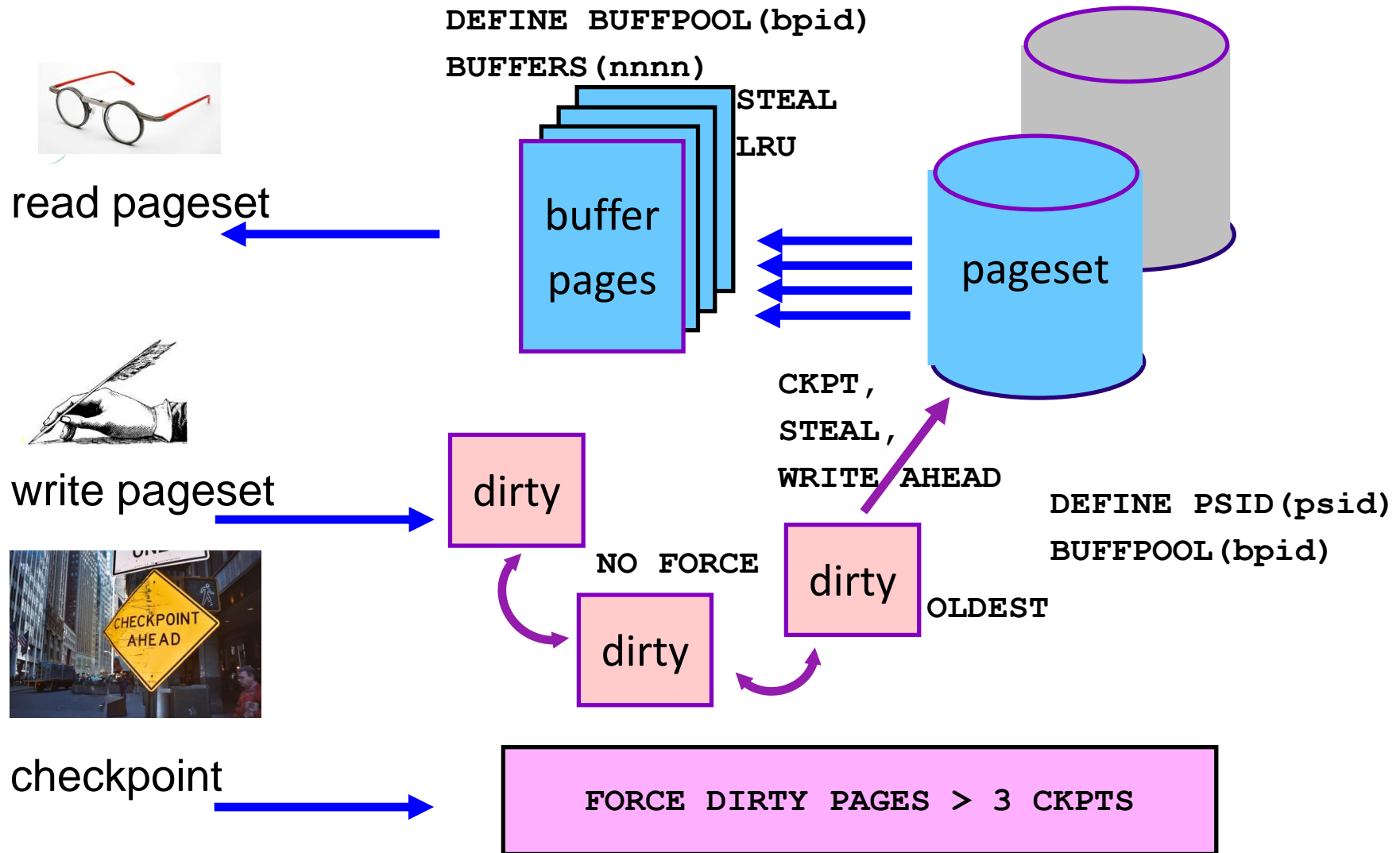
Controlling the MQI and MQSC - Message Manager



Controlling Messages and Objects - Data Manager



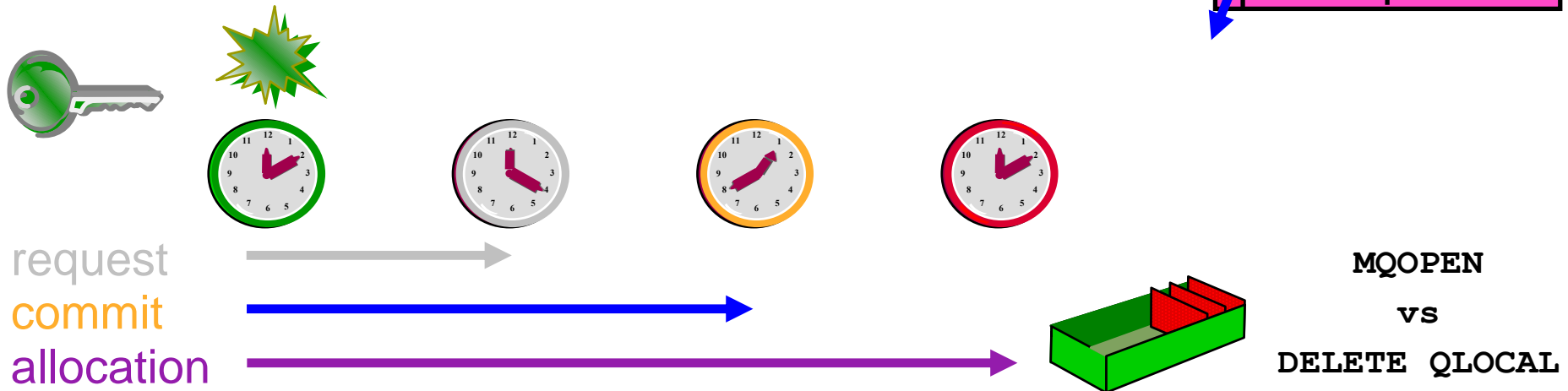
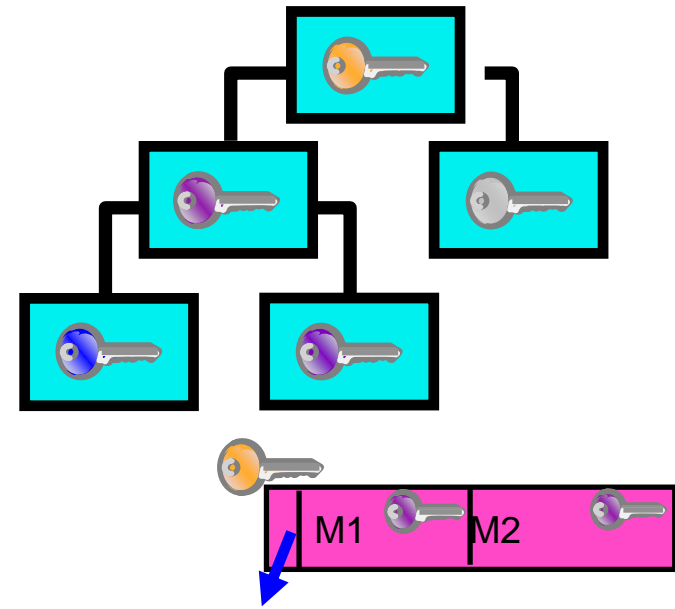
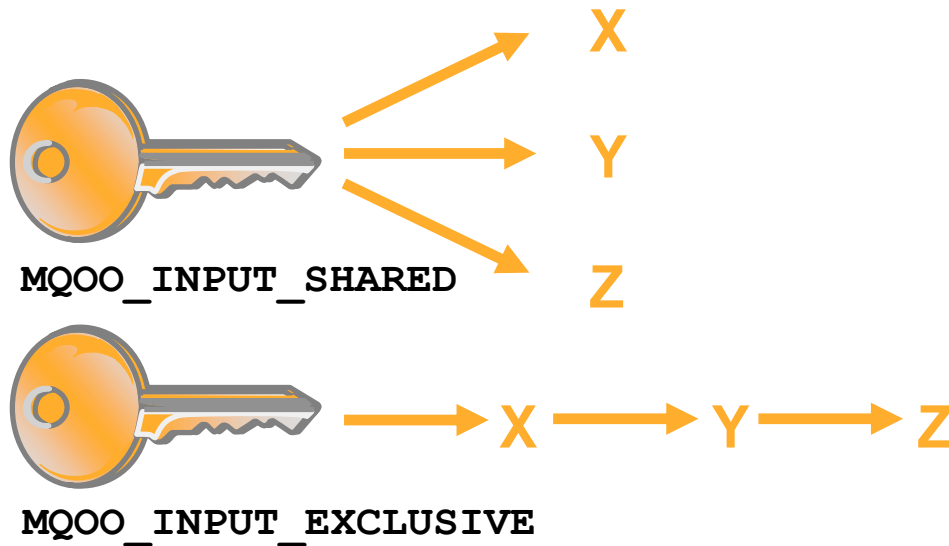
Buffer Manager – High Performance storage and retrieval







Providing Logging Interfaces - Log Manager

- Log read and write functions
- Log Shunting
- Multiple active log data sets and archive
- Archive inventory management
- Duplexed for reliability
- “Bootstrap” file
 - End of log location
 - Archive inventory
- Various Utilities




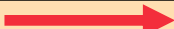
Concurrency and Isolation - Lock Manager



Scenario – Persistent MQPut to a Triggered Queue

| Application | Message Manager | Data Manager | Buffer Manager | Recovery Manager | Log Manager | Lock Manager |
|-------------|--|--|----------------|---|-------------|---------------|
| MQOPEN | | | | | | |
| |  | | | | | ACQUIRE LOCK |
| | | LOCATE QUEUE IN HASH TABLE | | | | |
| | SECURITY BASE NAME | | | | | |
| | ACQUIRE HANDLE | | | | | |
| MQPUT | | | | | | |
| | USE HANDLE | | | | | |
| | | LOCATE PAGE TO HOLD MSG | | | | |
| | | | BUFFER PAGE | | | |
| | |  | | START UR | LOG RECORDS | |
| | |  | | | LOG RECORDS | |
| | CHECK TRIGGER RULES | | | | | |
| MQCMIT | | | | | | |
| | | | |  | FORCE LOG | |
| | | | | | | RELEASE LOCKS |

Scenario - MQGet from a Queue

| Application | Message Manager | Data Manager | Buffer Manager | Recovery Manager | Log Manager | Lock Manager |
|-------------|--|--|----------------|---|-------------|------------------|
| MQOPEN | | | | | | |
| |  | | | | | ACQUIRE LOCK |
| | | LOCATE QUEUE IN HASH TABLE | | | | |
| | SECURITY | | | | | |
| | BASE NAME | | | | | |
| | ACQUIRE HANDLE | | | | | |
| MQGET | | | | | | |
| | USE HANDLE | | | | | |
| | | FIND MSG (INDEX / NEXT) | | | | |
| | | | BUFFER PAGE | | | |
| | |  | | START UR | LOG RECORDS | |
| | |  | | | LOG RECORDS | |
| MQCMIT | | | | | | |
| | | | |  | FORCE LOG | |
| | | | | | | RELEASE LOCKS |

Summary

- Delivers transactional messaging
 - Enables robust business applications
- Complex, but well organized
 - Adapters, Address spaces, Resource Managers
- Designed for throughput, availability and scalability
 - Logging, Buffering, Locking, Communications