IBM MQ for z/OS – Queue Sharing Groups - Help With Workload Skewing

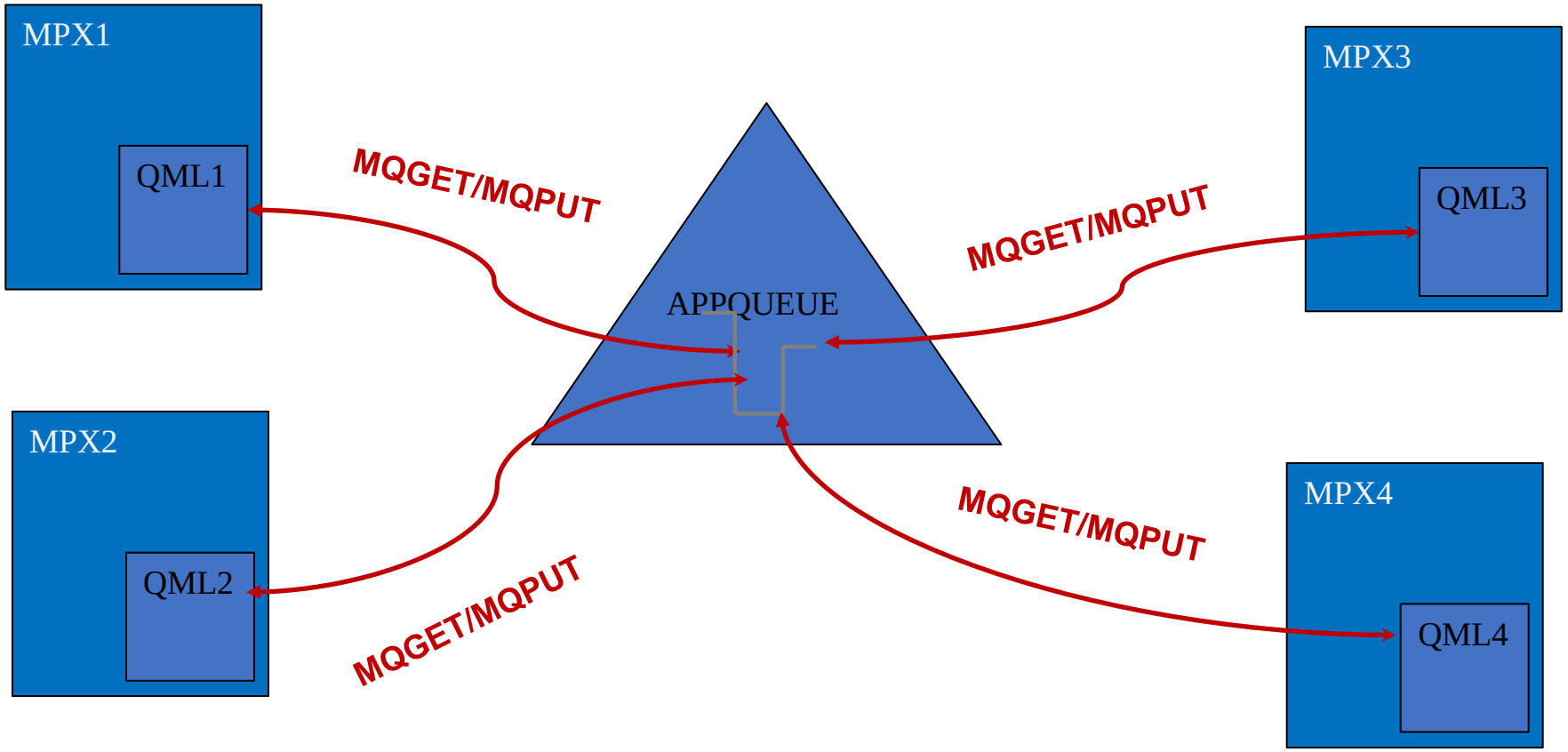
Dorothy Quincy
Dorothy.Quincy@ibm.com
IBM Washington Systems Center

Lyn Elkins
elkinsc@us.ibm.com
IBM Washington Systems Center

Agenda

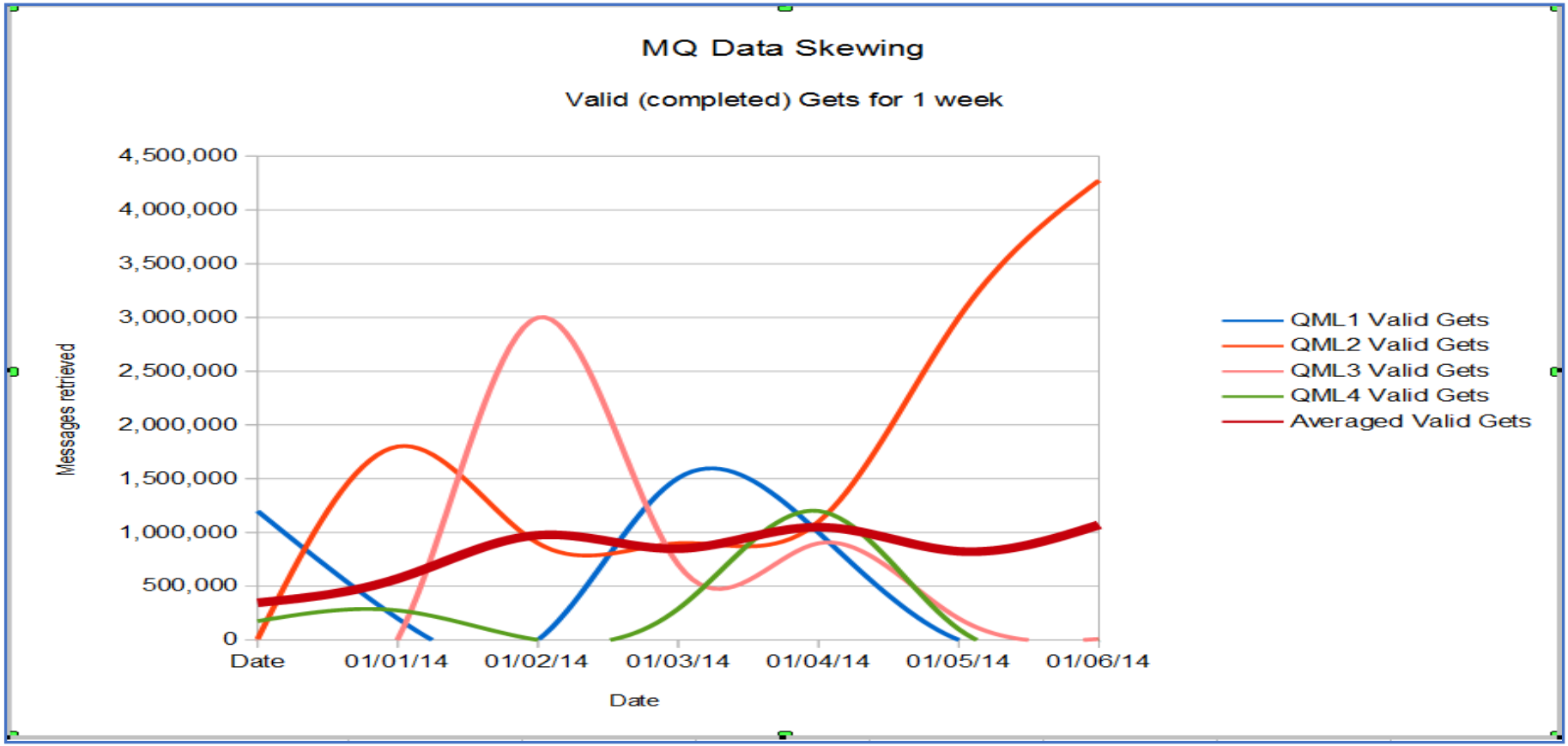
- What are shared queues – briefly
- What is workload skewing and why is it a problem?
  - Why am I talking about this again?
  - What are the symptoms and causes
    - Asymmetrical Sysplex
    - Connection Skewing
    - Put to Waiting Getter
    - Overnotification
  - Local' favoritism
- Mitigation Techniques:
  - Queue Manager Clustering
  - KEYRNOTIFYDELAY
  - Gateway queue managers
  - CICS CPSM options

Briefly – What is a Shared Queue



What is MQ Workload Skewing?

- Workload skewing is detected when MQ driven work, typically transactions, is not close to being evenly distributed across the queue managers.



Why am I talking about this again?

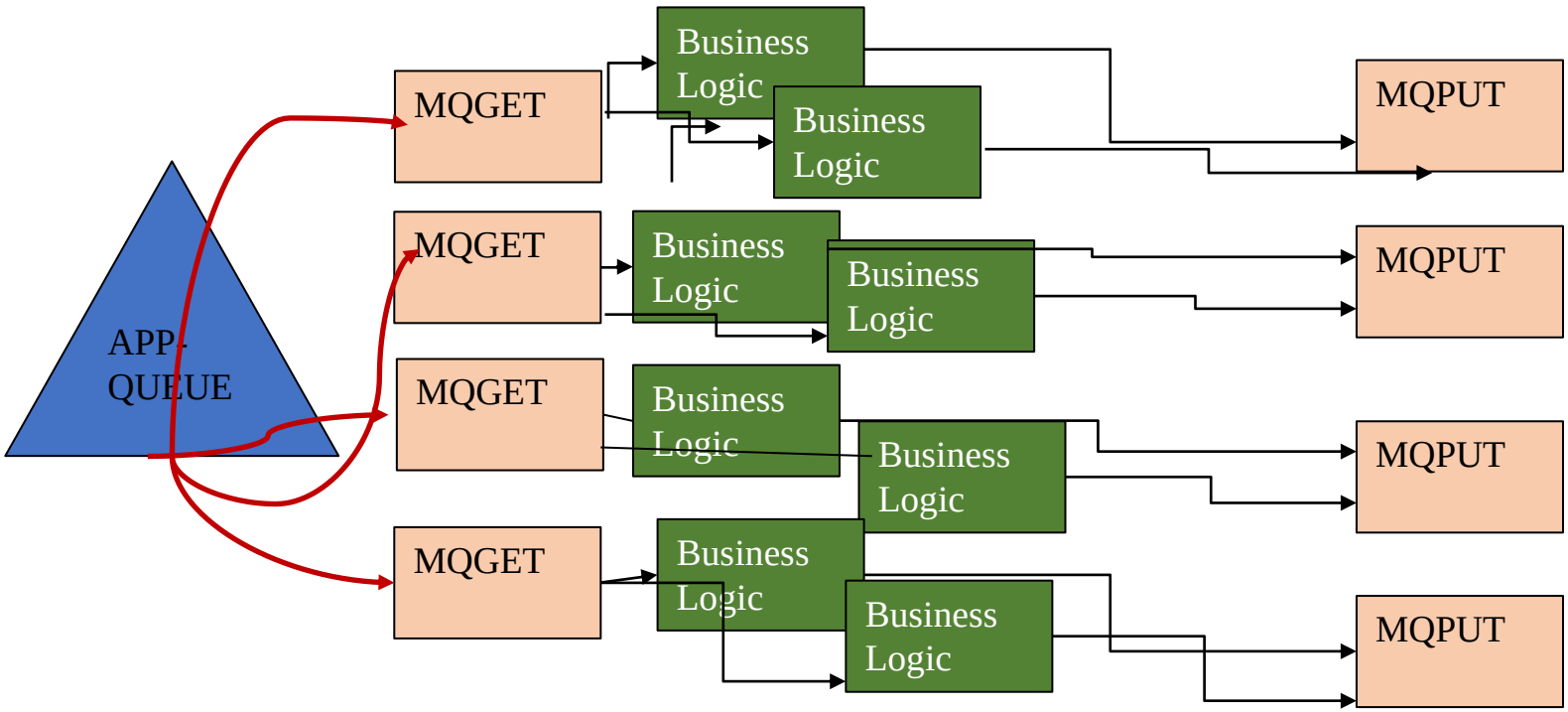
- New Hardware (z15s, z16s) has impact
- New Software (MQ V9.3+, z/OS 2.5, etc.) may have impact
- New configurations may have impact
  - Changes to the configuration can have unexpected benefits or impact
    - A recent case included a change to improve performance by upgrading the hardware and then downgrading it by changing from dedicated to shared engines
- You need to know what is happening to evaluate changes introduced
  - We have already tried to help some folks that didn't have 'before' data figure out what has changed.
    - We can guess, but we can never know!
  - Design and build some test cases to do before and after upgrade testing to compare behavior.
  - Please see my various rants about knowing what your baseline production environment 'looks' like from a performance and throughput perspective.

Why is MQ Workload Skewing a problem?

- It can be a cost problem
  - If the MLC ‘rolling average’ is taken from the LPAR that is heavily favored, usage pricing is not going to reflect reality
  - Technical solutions to this problem may prove to be less efficient overall - lower throughput, slower response
  - This was the first reason it was brought to our attention.
- It can be a Responsiveness problem
  - When all messages are consumed in one LPAR, it can lead to slowdowns (and shutdowns)
  - This was the second reason workload skewing was brought to our attention, it quickly became the #1 reason it continues to come our way.

And it can be a resource problem!

- Can cause increased capacity demands in downstream workload
  - Known to produce responsiveness problems
    - Overloading the message processing programs

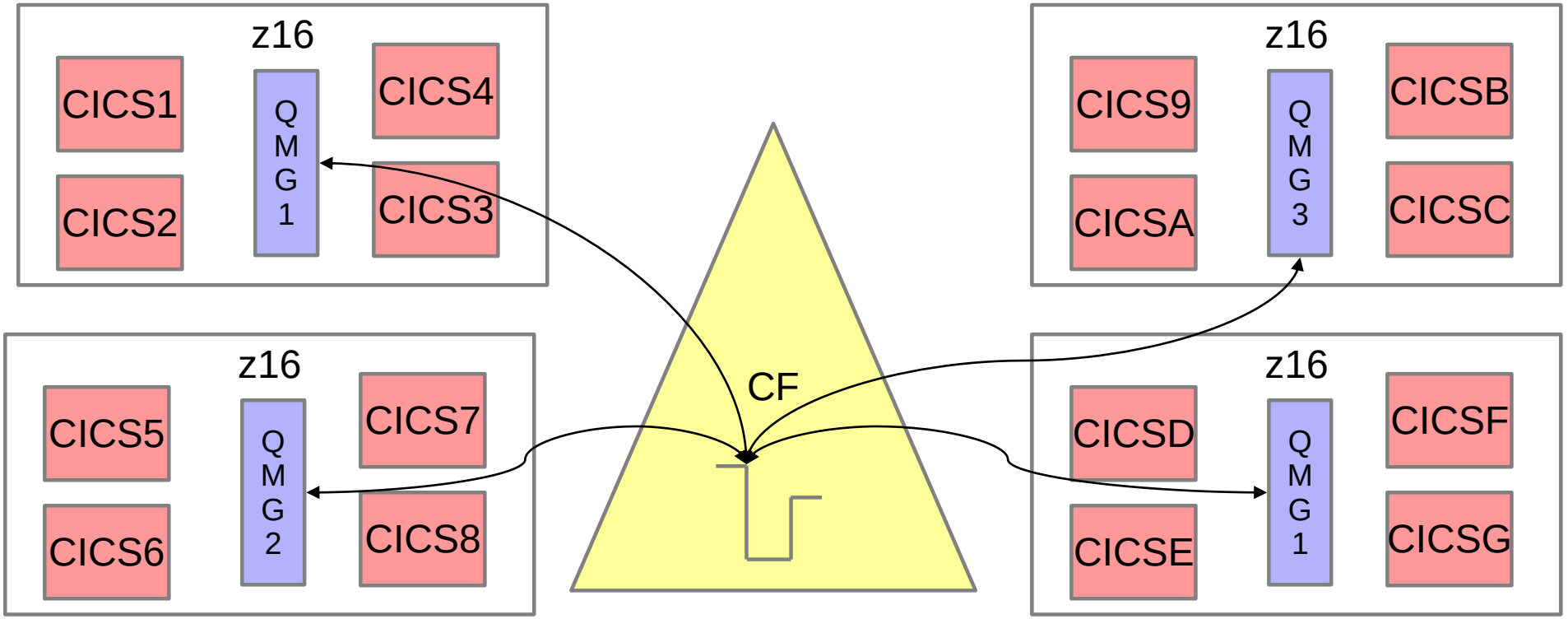


MQ Workload Skewing Causes

- **Workload skewing is often a result of the efficiencies of working locally**
  - z/OS, and all subsystems, try to process requests locally to take advantage of CPU efficiency
- **Workload skewing may be intentional**
  - Some applications may be affinity bound to an LPAR, but are using shared queues for the additional availability
  - Some applications are not yet Sysplex enabled
  - Software licensing agreements requiring LPAR restriction
  - Recommendation – know when this is the case and document.

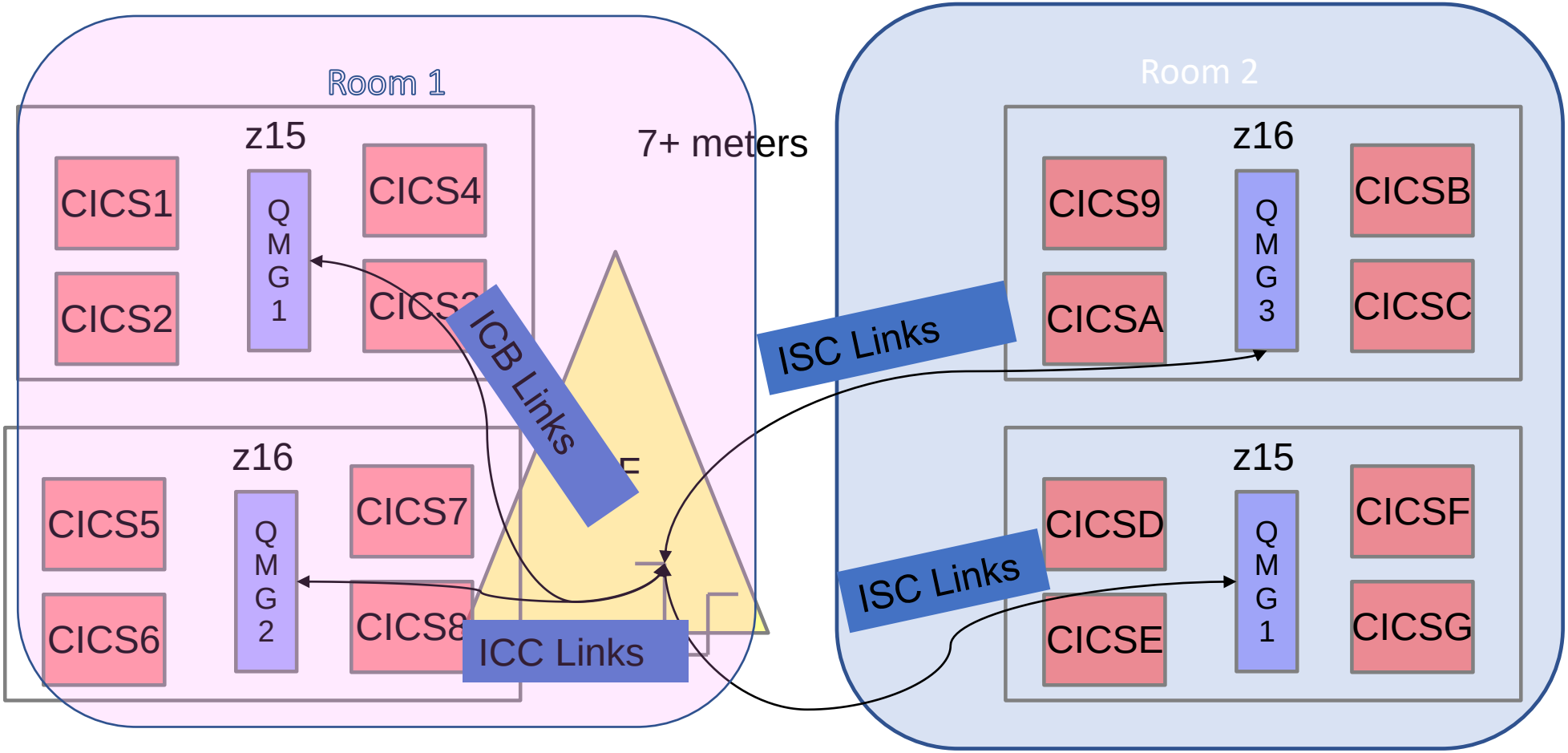
Asymmetric Sysplex – Hardware causes

- Asymmetric Sysplex
  - When the LPARs in the Sysplex are not equally weighted
    - Examples include:
      - Two LPARs have z16s, two have z16s
      - Two LPARs have 12 dedicated engines, two have 12 shared
      - All links to the CF are equally distant and are the same type



Physical causes of skewing – Links used to the CF

- Asymmetric Sysplex
  - Most common example - One LPAR is co-located with the primary coupling facility, the others are on different CECs
  - ICC and ISC links give much better service times than ICB



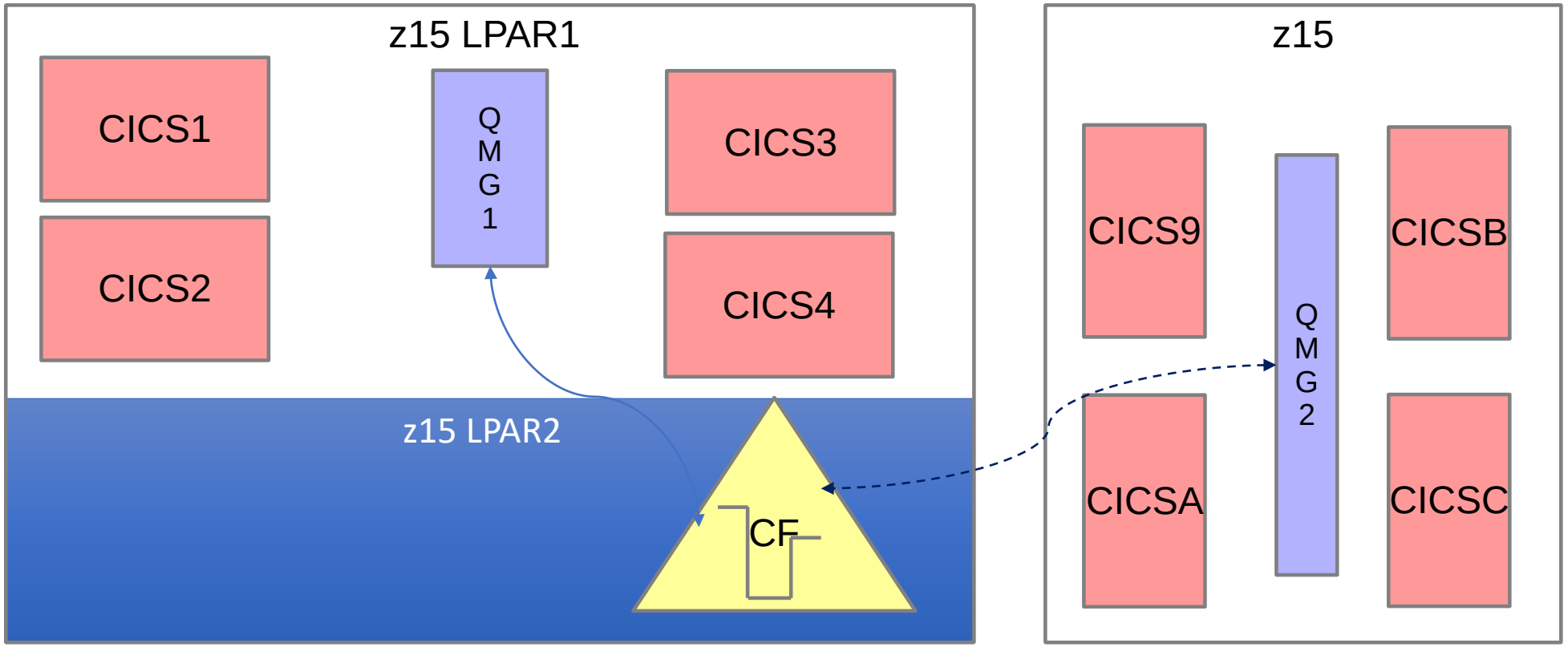
Physical Skewing – CF Activity Report

```
STRUCTURE NAME = QSGBUSER          TYPE = LIST    STATUS = ACTIVE
          # REQ    -------------- REQUESTS -------------   -------------- DE
SYSTEM    TOTAL                 #     % OF  -SERV TIME(MIC)-    REASON     #    %
NAME      AVG/SEC               REQ    ALL     AVG    STD_DEV              REQ    R

MPX1       295K   SYNC        295K   26.9     4.3       1.2     NO SCH     0     0
          492.1   ASYNC         0     0.0     0.0       0.0     PR WT      0     0
                  CHNGD         0     0.0  INCLUDED IN ASYNC    PR CMP     0     0
                  SUPPR         0     0.0                       DUMP      0     0

MPX2       802K   SYNC        802K   73.1    17.8       2.5     NO SCH     0     0
          1339    ASYNC         0     0.0     0.0       0.0     PR WT      0     0
                  CHNGD         0     0.0  INCLUDED IN ASYNC    PR CMP     0     0
                  SUPPR         0     0.0                       DUMP      0     0
```

- We (the WSC) look at the CF Activity report – usually before the MQ Statistics when looking at shared queue usage
- In the example shown above it is easy to see that the MPX2 LPAR is getting a much longer service time (almost 4 times!) than the MPX1 LPAR and that MPX2 is making many more requests.
  - In this particular case, this exposed some internal workload skewing that was not apparent to the customer - except that they were missing SLAs consistently!

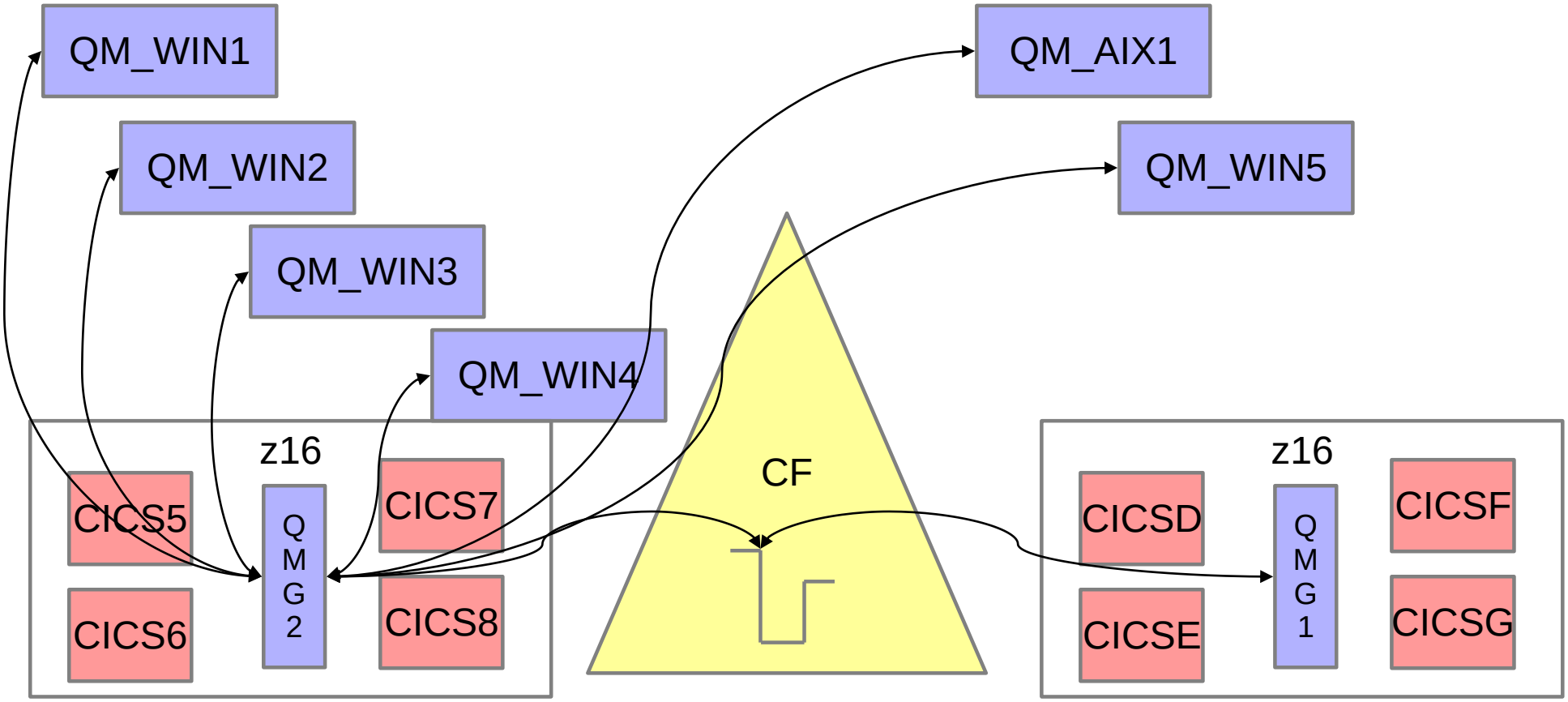
MQ Workload Skewing Causes - Hardware

- Location of the Coupling Facility
  - When the coupling facility is internal, LPARs on the same CEC tend to get faster response
  - When the coupling facility is external and one LPAR has more, faster, or less heavily used links it will get faster service



Connection Skewing

- Connection skewing may be historical
  - Hard-coded connections to specific queue managers
- Connection skewing may be the result of a queue manager outage
  - Connections to a QSG are routed to available queue managers

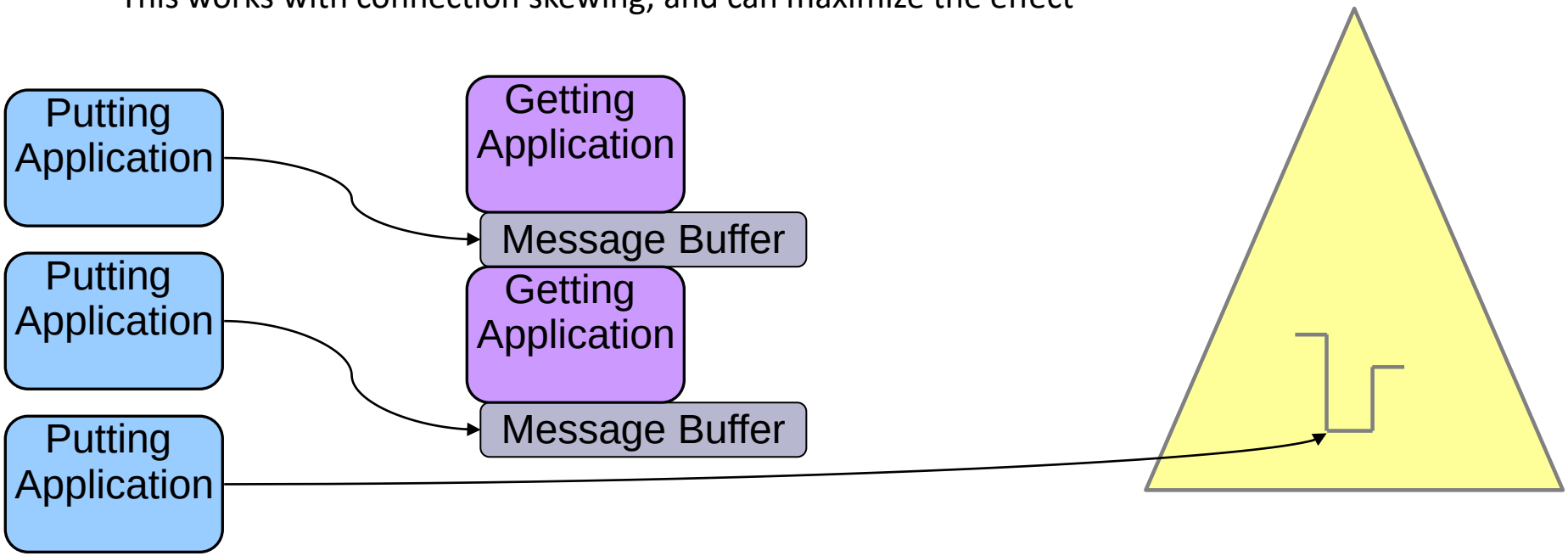


‘Downstream’ consequences

- We’ve talked about the MLC impact
- Resource use
  - Not every queue manager is sized to absorb the entire workload
  - Log impact of skewing has been seen
    - Rapid Log switches due to heavier workload – increasing I/O and CPU costs
  - Bufferpool/Pageset impact
    - Filling the bufferpool, forced into I/O
  - SMDS impact
    - One queue manager in QSG gets all offloaded messages

MQ Workload Skewing Causes

- Put to waiting getter
  - In V6 a performance feature was added called ‘put to waiting getter’
  - If a local put, from an application or message channel agent, is done and there is a getting application waiting the message is moved directly to the getting applications buffer
    - There is no posting to a shared queue
    - There is no notification to other available waiting applications
    - The CPU savings can be substantial
    - This works with connection skewing, and can maximize the effect



Put to Waiting Getter – SMF

- This shows messages flowing across a channel taking advantage of P2WG

Base_Name	CF Struct	Total_Valid_Gets	Total_Bytes_Put	Total_Valid_Puts	Total_Put2_Waiting_Getter	Puts not to Waiting Getter
SYSTEM.QSG.CHANNEL.SYNCQ	CSQSYSAP	0	0	0	0	0
SHARED.INPUT.QUEUE	APP1	0	4501092223	2095814	2012394	83420

- The CPU comparison shows why it can be a good thing

BASE_NAME	VALID_PUTS	PUT_ELAPSED_TIME	PUT_CPU_TIME	PUT2_WAITING_GETTER	Average PUT ET	Average PUT CT
QLOCAL.PUT2WG	14879	127753	117956	14793	8.59	7.93
QLOCAL.NO.PUT2WG	41547	1025028	1010038	0	24.67	24.31

- The CPU costs can be 3 times as high!

MQ Workload Skewing Causes

- Local Favoritism
  - When a message is posted to a shared queue, the queue manager where the message is put is typically notified FIRST about the availability.
  - Normal processing by XCF, taking advantage of the efficiency of local processing.

Over-notification – Triggering Avalanche

- Triggering processes on shared queues can contribute to workload skewing:
  - If using shared initiation queues it can be worse
    - We (the WSC) have seen an 8 way QSG, where all 8 trigger messages were consumed by 1 queue manager due to hardware skewing.
  - Simply, the number of trigger messages created and consumed will vary and if there is any other type of imbalance it can become worse.
  - Over-notification can also cause excessive CPU consumption:
    - For example, in an 8-way queue manager using shared triggered and init queues
      - As many as 8 trigger messages are created on every ‘0 to non-zero’ transition
      - If 8 instances are triggered and there are fewer than 8 to consume, some (or most) will get 2033
      - The cost of initiating the excess transactions adds up, especially when applications are not well behaved

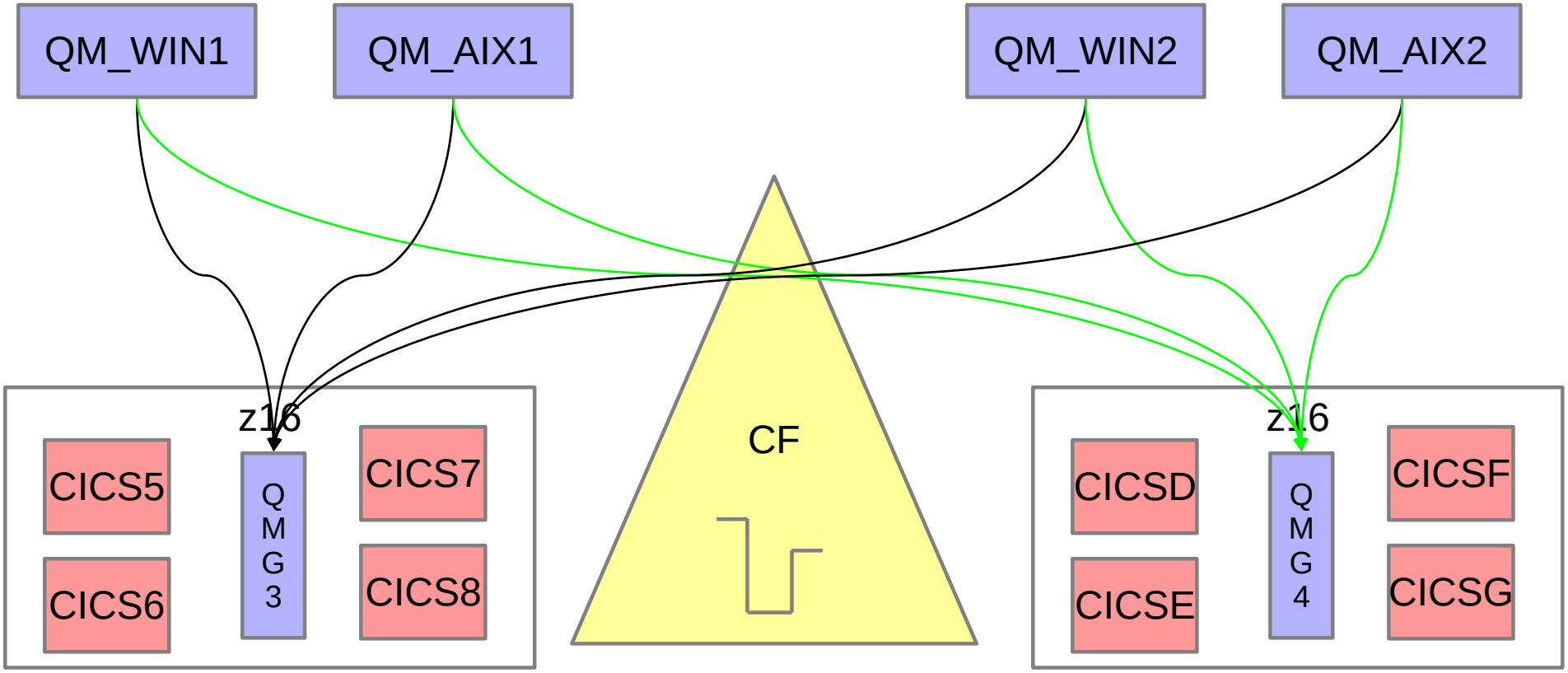
Skewing Mitigation Techniques

- Queue Manager Clusters
  - Clusters provide workload balancing across queue managers
  - Works with shared queues to distribute message ‘puts’ across queue managers in the QSG
- Connection skewing mitigation
  - Gateway queue managers
  - Re-driving connections
- CPSM mitigation

Queue Manager Clustering

•When messages are not bound to a specific queue manager (‘bind not fixed’), the messages are routed evenly across the receiving queue managers

- Black arrows show the first message put to the clustered queue
- Green arrows show the second message

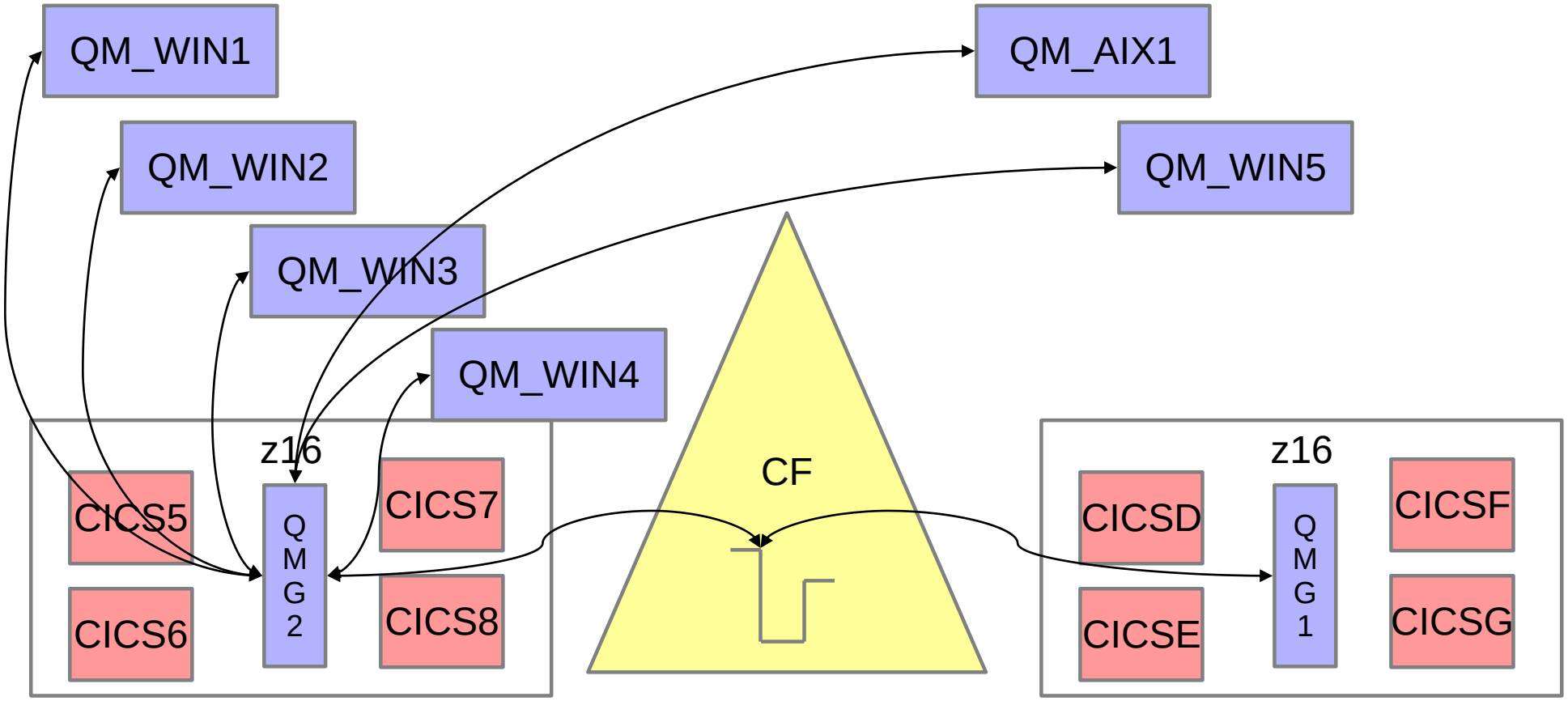


Connection Skewing Mitigation

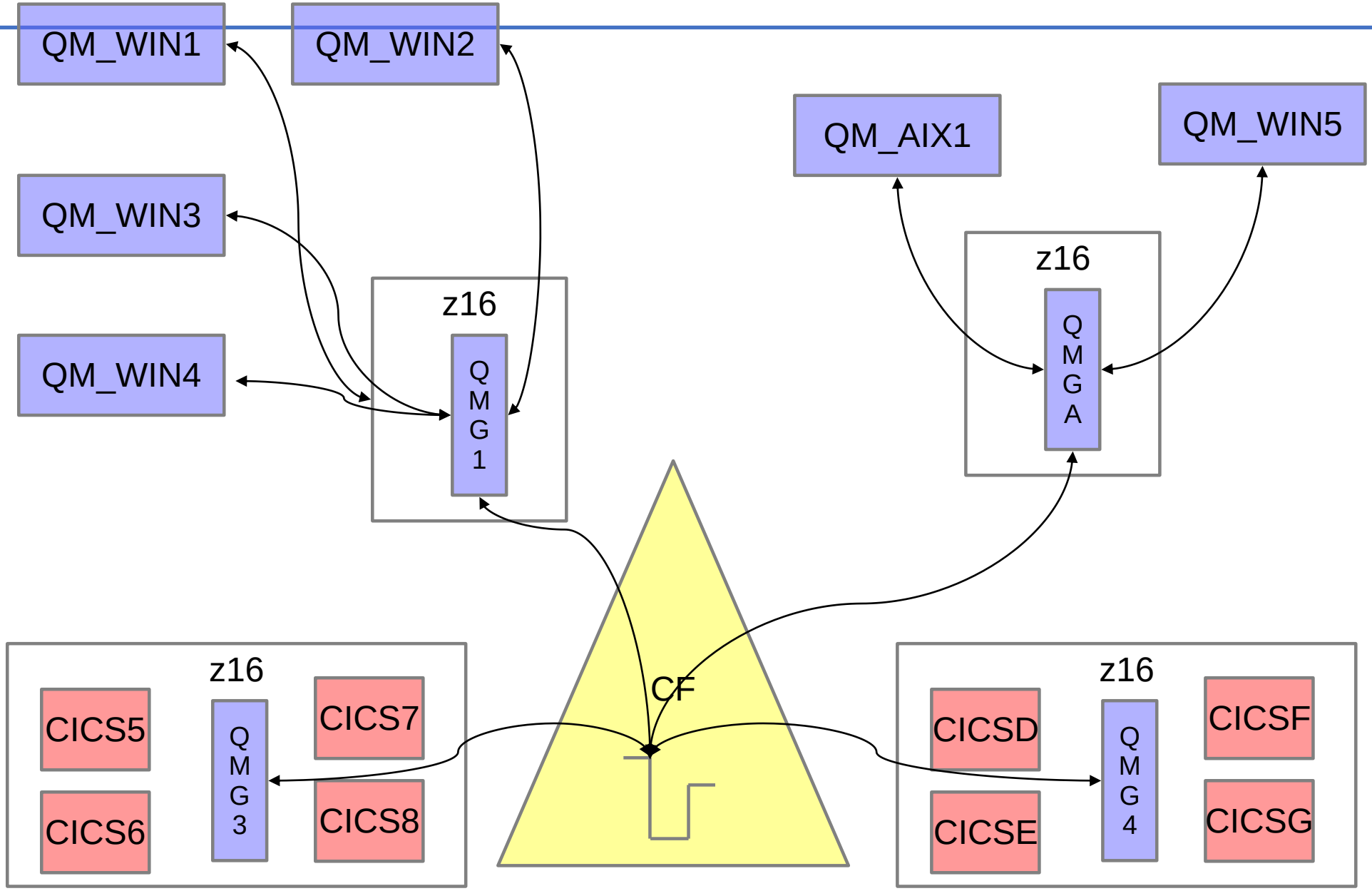
- The slides that follow outline two mitigation techniques for connection skewing:
  - Gateway queue managers
  - Re-driving connections

Connection Skewing – No Gateway queue managers

- When external queue managers or clients are passing work directly to application hosting queue managers, every attempt is made to process the work locally
- Environments that use gateway queue managers into the Queue Sharing group often eliminate connection skewing.

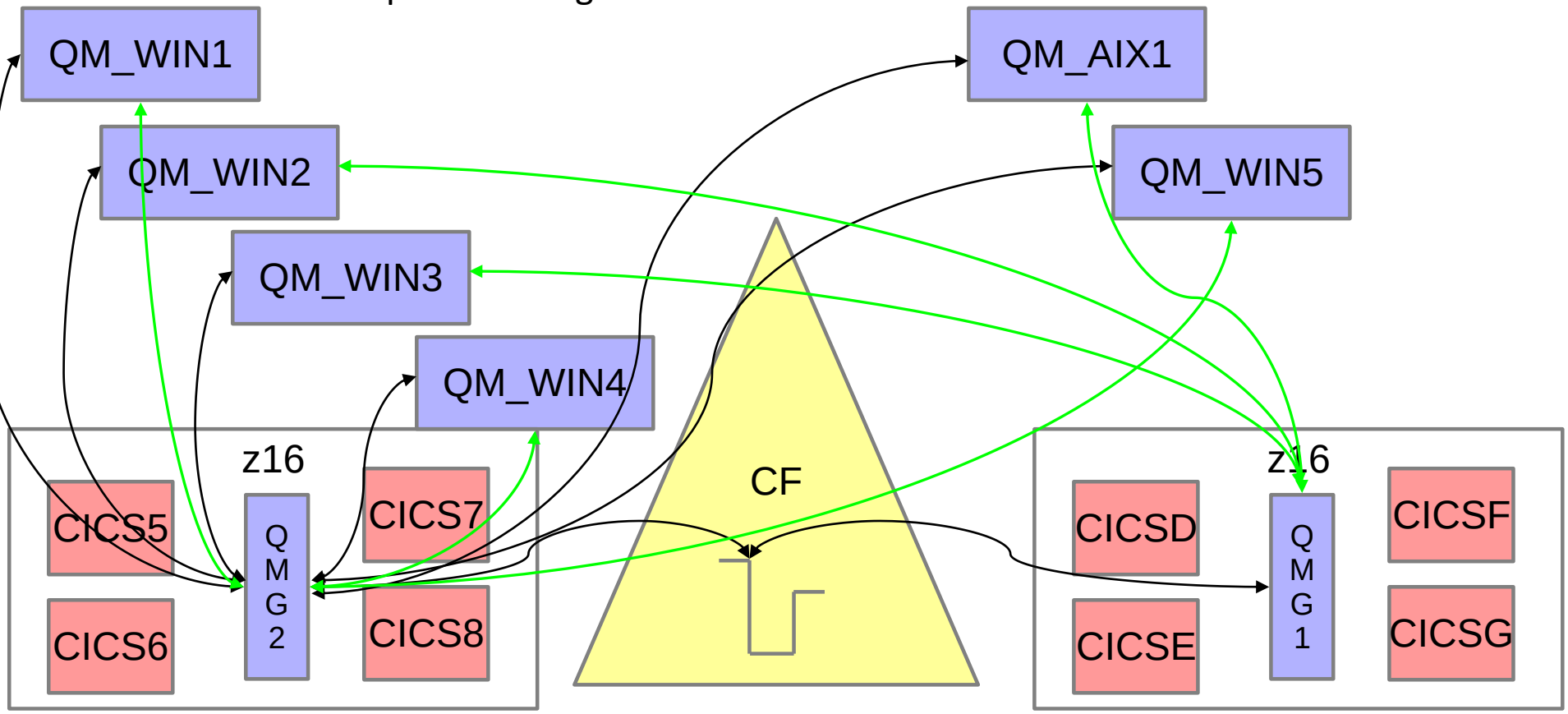


Gateway queue managers – the mitigation



Re-driving Connections

- When a queue manager is unavailable, inbound connections can get skewed to the other queue manager(s) in the group.
  - This is normal availability processing!
  - Once a connection is live and active, no attempt is made to balance the connections once all the queue managers are available.



Trigger Avalanche Mitigation

- KEYRNOTIFYDELAY (KRND) is a Coupling Facility attribute designed to help reduce over notification
- If used, when a trigger event occurs only one queue manager is initially notified
  - If the timer expires (the delay) and the queue is not empty; all other queue managers are notified.
  - For workloads that have sporadic delivery patterns, this can save a lot of CPU.
- Apply PH44368

CPSM Highlights

Solution uses proven technology for CPSM routing:

- Each TOR/QOR uses link-neutral goal algorithm
  - Selects target AOR based on AOR load and health
  - Does not “prefer” local (= same LPAR) AORs
  - Even distribution across AORs, but …
  - … responds to transient load/health variation
- XCF MRO for “remote” STARTs or LINKs
  - High-performance System z sysplex technology
  - Uses coupling facility (CF) instead of TCP/IP stack
- Sysplex-optimised workload routing
  - Highly responsive to transient variations
  - Uses CF to maintain current status for AORs

Continuous operation and high availability through IBM MQ shared queues:

- “Glitchless” recovery from region/LPAR/CEC outage
- “Instant” redistribution of workload
- In-flight messages backed-out, restart in another CICS region

High throughput:

- Exploits all available capacity
- Highly responsive to transient spare capacity

MQ Workload Balance Summary

- MQ is a message delivery system, it does not try to balance workload or distribute workload
- Balancing the workload is attempting a technical solution for what is often a pricing problem
  - Beware spending a lot of effort for a solution to a temporary problem as well!
  - Turning off performance improvements like put to waiting getter will impact all applications, not just the skewed ones
- There are some mitigation techniques that can help the overall environment
  - Clustering!
  - Gateway queue managers
  - Using CPSM to make appropriate routing decisions

Questions & Answers