Where is my message?

Matt Leming
lemingma@uk.ibm.com

Agenda – the MQ Toolbox

- What's connected?
- Are messages flowing?
- Where are messages going?
- What are the apps doing?
- How can I look back in time?

NB: Capabilities differ in form between Distributed and z/OS, this presentation is Distributed focused

Presentation contains MQSC example commands and output. Your own admin tools will vary by task, situation and preference. Tasks can be performed in PowerShell, MQ Explorer etc. as desired.

What's connected?

DISPLAY CONN

All platforms

DISPLAY CONN(*) TYPE(HANDLE) ALL

```
AMQ8276: Display Connection details.
   CONN(577C425321295301)
   EXTCONN(414D5143474154455741593120202020)
   TYPE(HANDLE)

   OBJNAME(WLMMDB.REQUEST)                  OBJTYPE(QUEUE)
   ASTATE(NONE)                             HSTATE(INACTIVE)
   OPENOPTS(MQOO_OUTPUT,MQOO_FAIL_IF_QUIESCING)
   READA(NO)

   OBJNAME(SENDINGAPP.REPLY)                OBJTYPE(QUEUE)
   ASTATE(ACTIVE)                           HSTATE(ACTIVE)
   OPENOPTS(MQOO_INPUT_SHARED,MQOO_INQUIRE,MQOO_SAVE_ALL_CONTEXT,MQOO_FAIL_IF_QUIESCING)
   READA(NO)
```

Use CONN to match TYPE(CONN) and TYPE(HANDLE) records

TYPE(HANDLE) records let you find applications by the objects they access.
See all open handles for an app in one place, unlike DIS QSTATUS records

DISPLAY CONN(*) ALL

```
AMQ8276: Display Connection details.
   CONN(577C425321295301)
   EXTCONN(414D5143474154455741593120202020)
   TYPE(CONN)
   PID(9740)                                TID(185)
   APPLDESC(WebSphere MQ Channel)           APPLTAG(jms/GATEWAY1_CF)
   APPLTYPE(SYSTEM)                         ASTATE(NONE)
   CHANNEL(WAS.CLIENTS)                     CONNAME(127.0.0.1)
   CONNOPTS(MQCNO_SHARED_BINDING)           USERID(pbroad)
   UOWLOG( )                                UOWSTDA(2014-04-08
   UOWSTTI(13.24.00)                        UOWLOGDA( )
   UOWLOGTI( )                              URTYPE(XA)
   EXTURID(XA_FORMATID[DSAW]
XA_GTRID[00000145414B8AB40000000104DF48FC0001020304050607080900
XA_BQUAL[00000145414B8AB40000000104DF48FC0001020304050607080900
0000000001])
   QMURID(0.7940075)                        UOWSTATE(ACTIVE)
```

Channel name + IP help identify client apps.

MQ V7.5 and later JMS clients can supply an application name in the CF

Long running UOW information.
XID can be tied up with app server txn timeout

DISPLAY CHSTATUS

```
DISPLAY CHSTATUS(*) ALL
```

```
AMQ8417: Display Channel Status details.
   CHANNEL(WAS.CLIENTS)                    CHLTYPE(SVRCONN)
   BUFSRCVD(17)                            BUFSSENT(13)
   BYTSRCVD(2296)                          BYTSSENT(2456)
   CHSTADA(2014-04-08)                     CHSTATI(15.26.59)
   COMPHDR(NONE,NONE)                      COMPMSG(NONE,NONE)
   COMPRATE(0,0)                           COMPTIME(0,0)
   CONNAME(127.0.0.1)                      CURRENT
   EXITTIME(0,0)                           HBINT(5)
   JOBNAME(0000260C000000B9)               LOCLADDR( )
   LSTMSGDA(2014-04-08)                    LSTMSGTI(15.26.59)
   MCASTAT(RUNNING)                        MCAUSER(pbroad)
   MONCHL(OFF)                             MSGS(6)
   RAPPLTAG(jar)                           SSLCERTI(CN=ExampleCA,O=Example)
   SSLKEYDA( )                             SSLKEYTI( )
   SSLPEER(SERIALNUMBER=63:43:FD:D6,CN=ExampleApp1,O=Example)
   SSLRKEYS(0)                             STATUS(RUNNING)
   STOPREQ(NO)                             SUBSTATE(RECEIVE)
   CURSHCNV(1)                             MAXSHCNV(1)
   RVERSION(00000000)                      RPRODUCT(MQJM)
```

Check suitable heartbeats are negotiated

See SSLPEER information not in DIS CONN

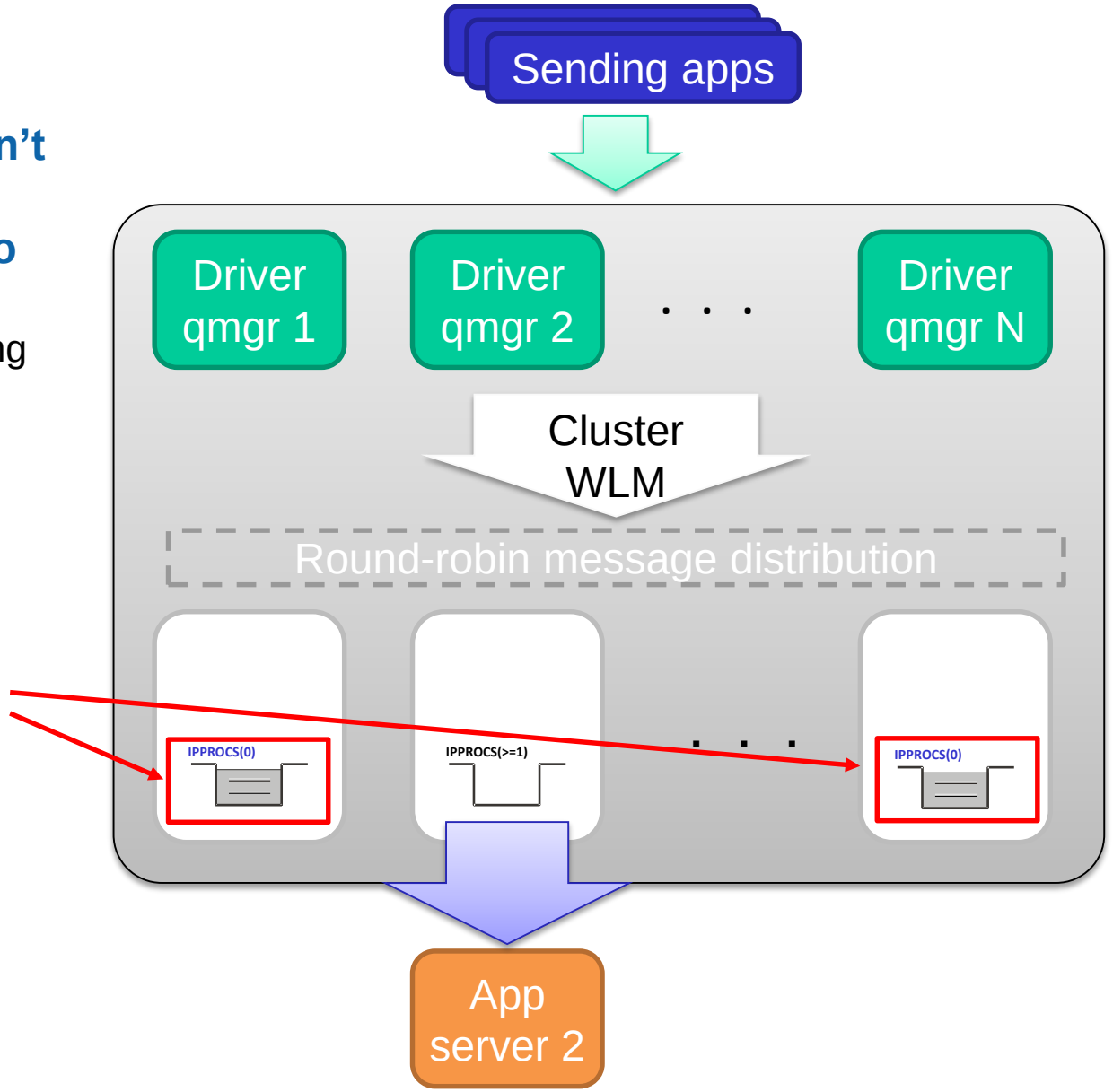
JOBNAME contains PID (except z/OS): 0x260C = PID(9740)
On Linux/UNIX (not Win) TID matches CONN: 0xB9 = TID(185)

Note that multiple CONN might share one SVRCONN channel instance

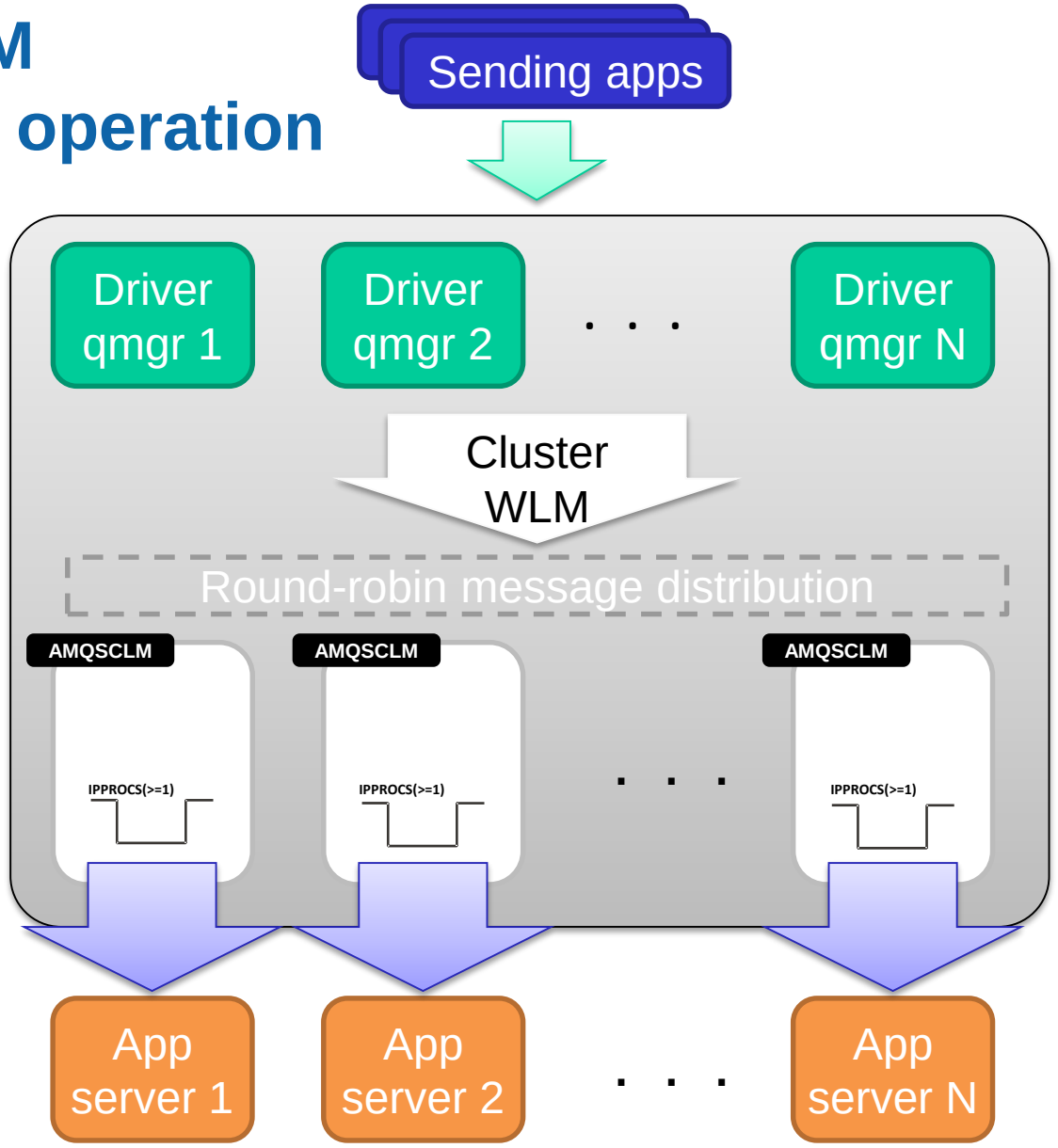
Rerouting messages if no-one is connected

Cluster WLM

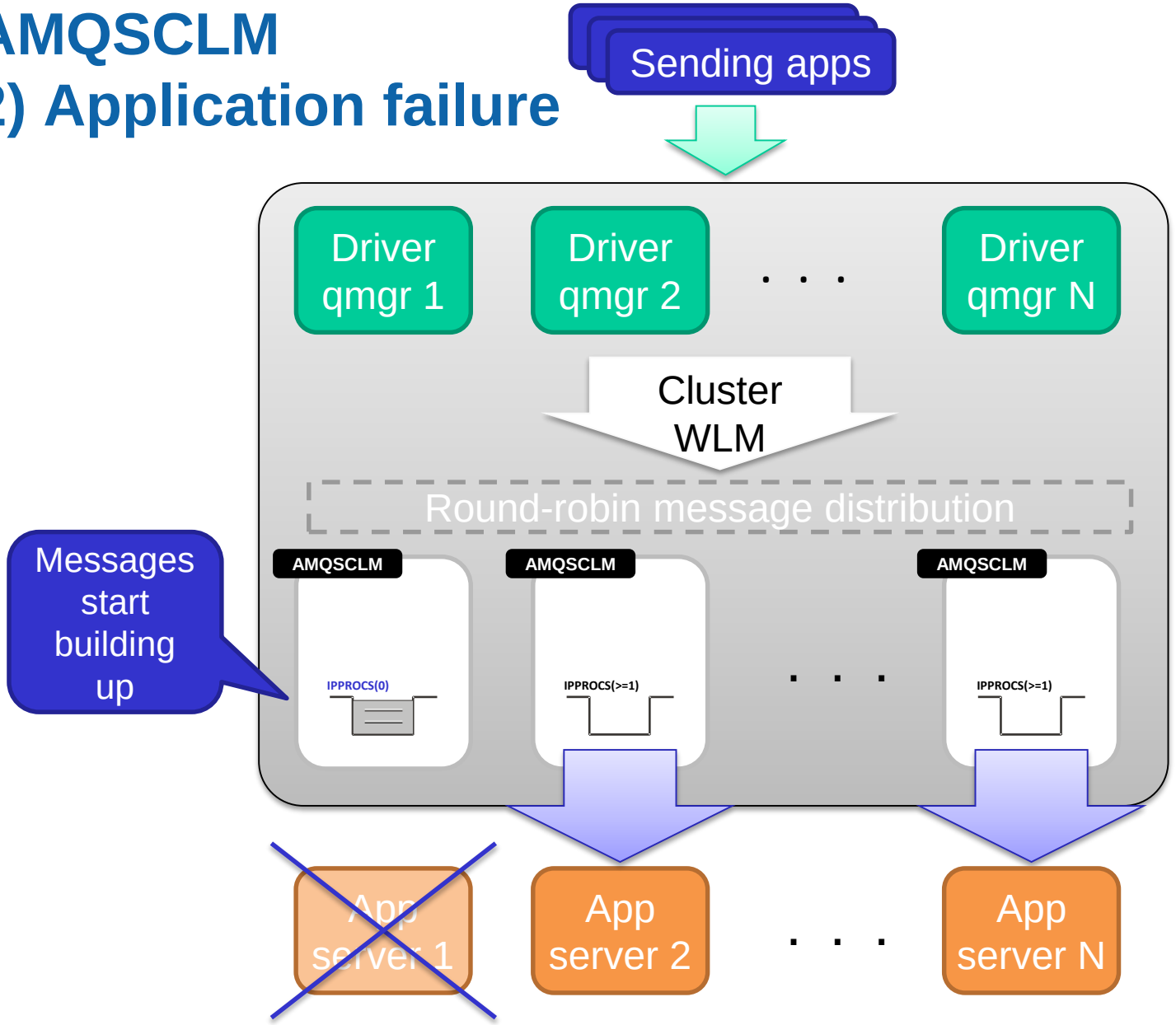
- **Cluster WLM algorithm doesn't check whether consuming applications are connected to cluster queues**
  - I.e. whether something is getting messages from them
- **You need to ensure that applications are consuming from all instances of a clustered queue to prevent messages building up**
- **However, there is an alternative: AMQSCLM**



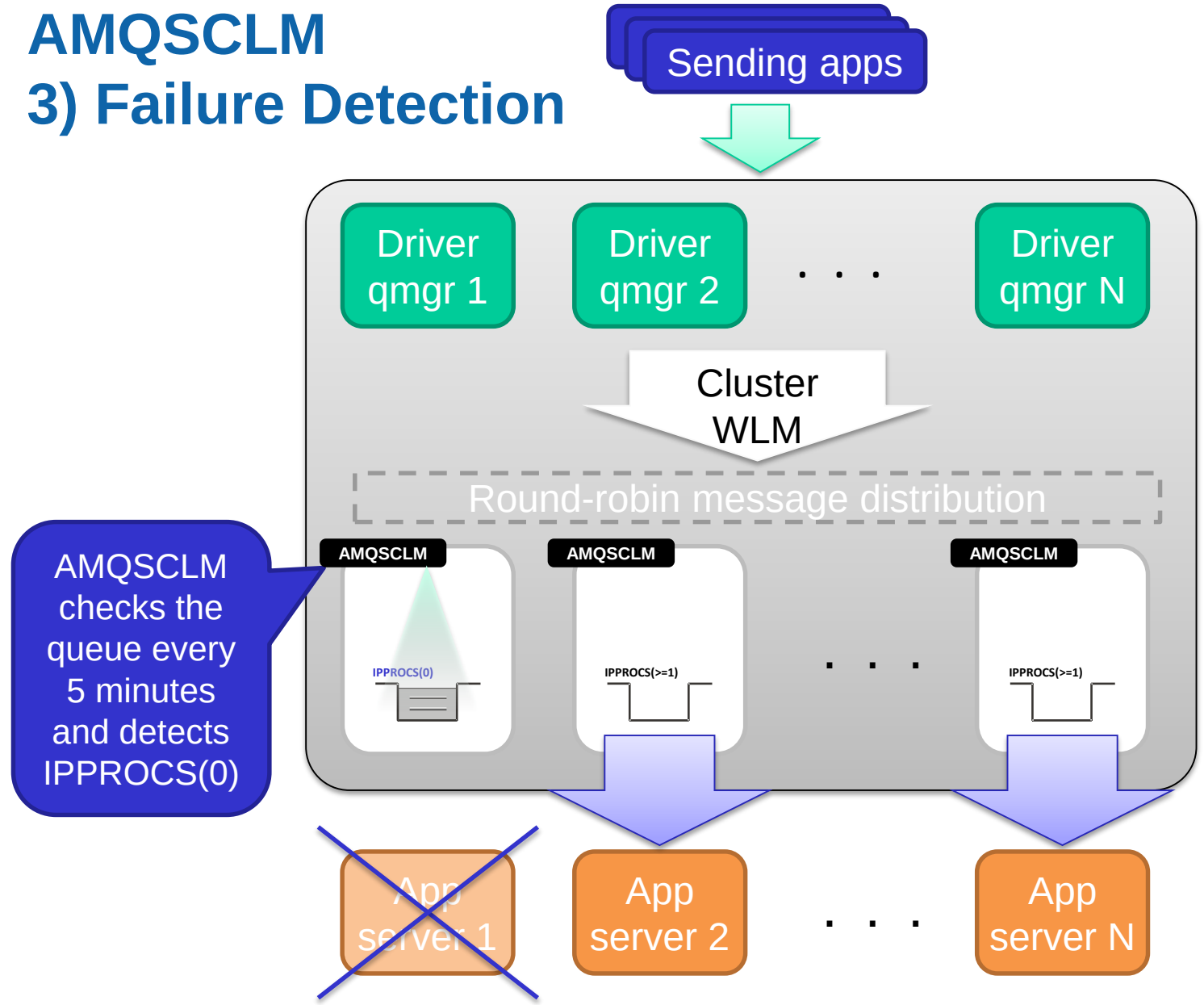
AMQSCLM
1) Normal operation



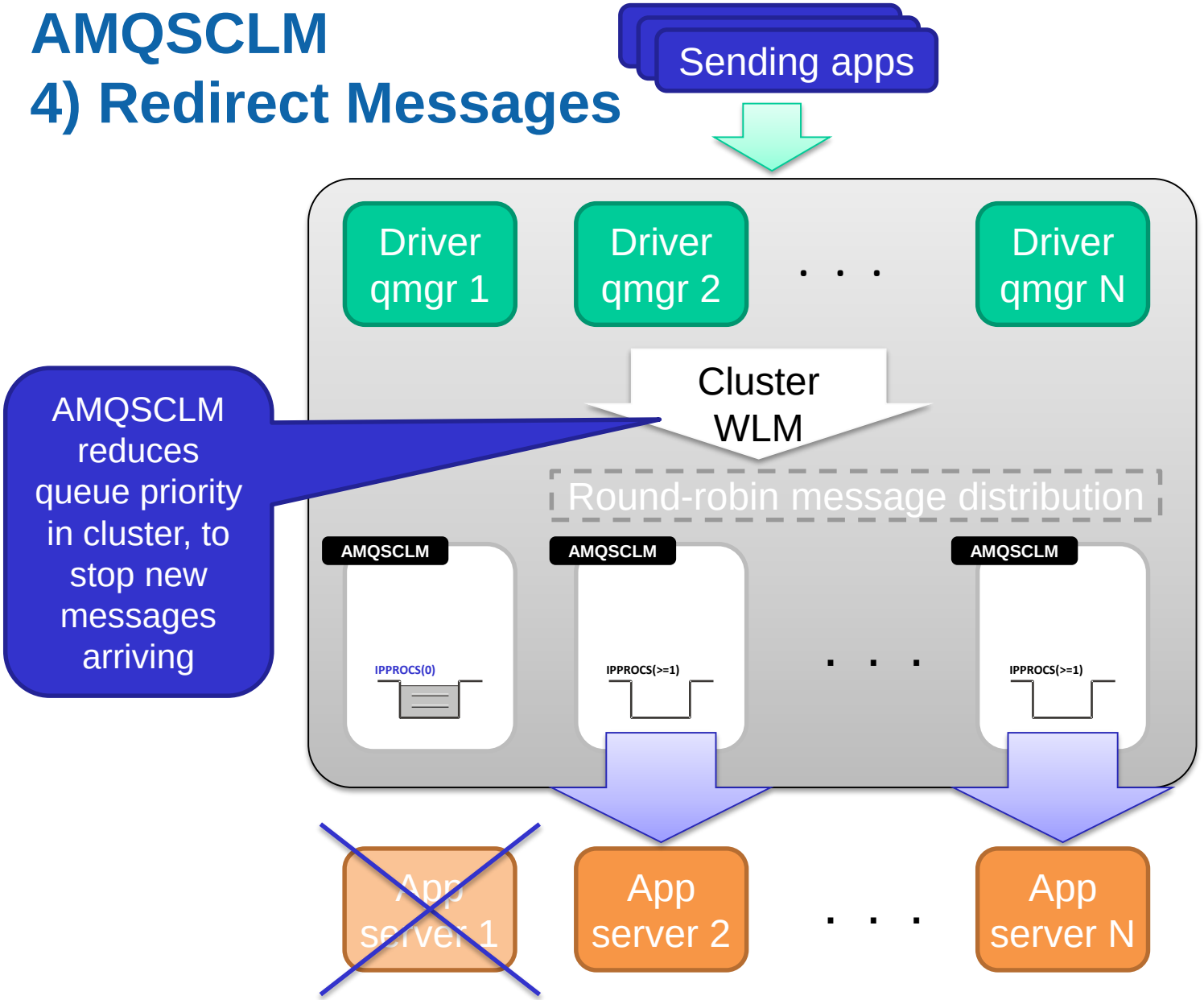
AMQSCLM
2) Application failure



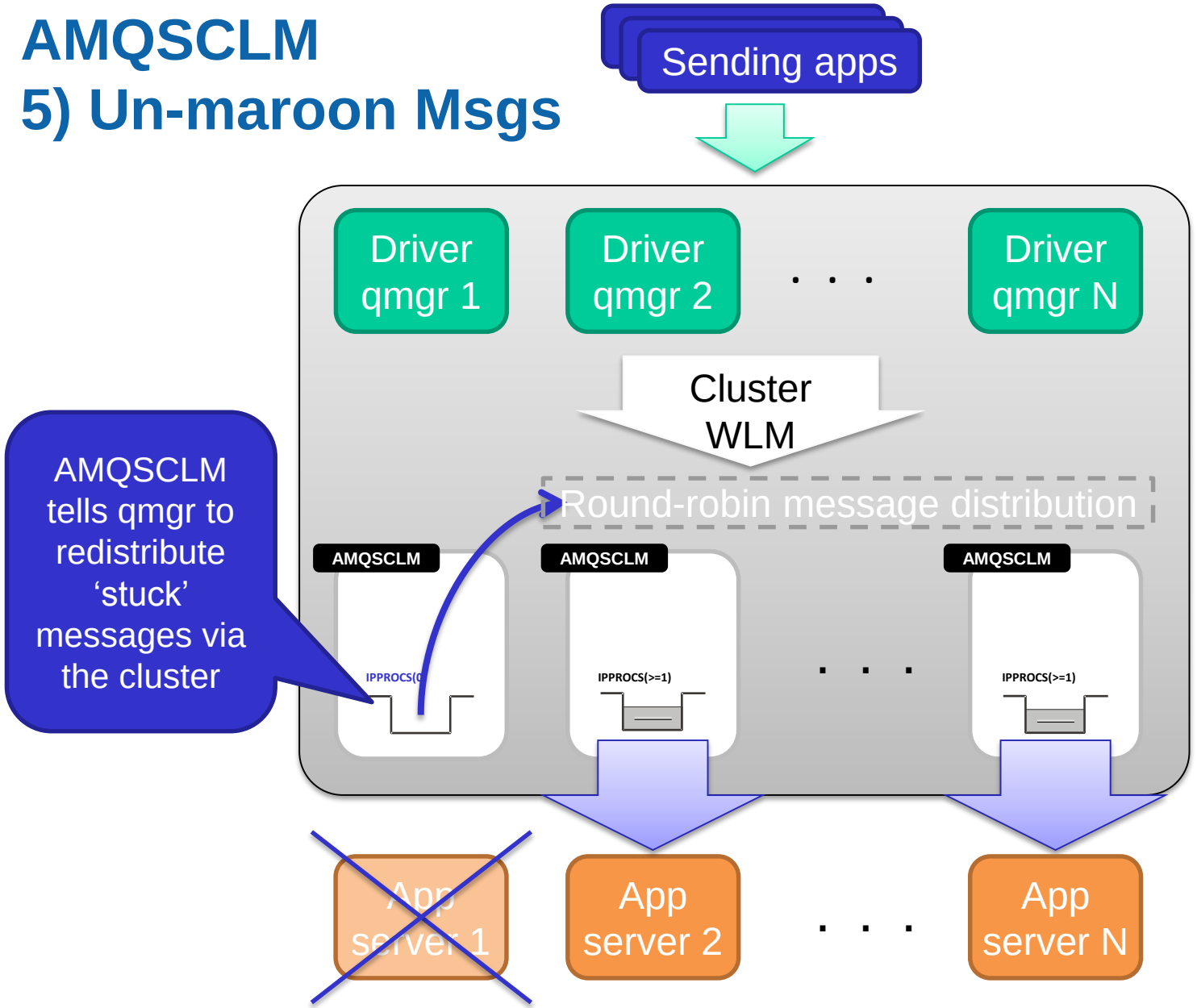
AMQSCLM
3) Failure Detection



AMQSCLM
4) Redirect Messages



AMQSCLM
5) Un-maroon Msgs (optional)



AMQSCLM summary

- **The cluster queue monitoring sample program (AMQSCLM)**

- **Shipped with the product as a sample**
  - Precompiled
  - Source code
  - Not shipped on z/OS, but distributed source code can be compiled and used on z/OS

- **More information here:**
  http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.dev.doc/q024620_.htm

Are messages flowing?

Real-time/online monitoring – queues

- **Set detail level for queue manager. Override for individual queues**

```
ALTER QMGR MONQ(MEDIUM)
ALTER QLOCAL(QUEUE1) MONQ(HIGH)
ALTER QLOCAL(QUEUE2) MONQ(OFF)
```

```
DIS QSTATUS(QUEUE1) ALL
```

- **Gives live view of application responsiveness**

```
AMQ8450: Display queue status details.
   QUEUE(QUEUE1)                   TYPE(QUEUE)
   CURDEPTH(16)                    IPPROCS(3)
   LGETDATE(2014-04-08)            LGETTIME(17.05.59)
   LPUTDATE(2014-04-08)            LPUTTIME(17.12.16)
   MEDIALOG( )                     MONQ(HIGH)
   MSGAGE(112)                     OPPROCS(5)
   QTIME(10101414, 10101414)       UNCOM(NO)
```

Without MONQ you only get the depth and how many handles are open

Timestamps of last PUT/GET to check for recent activity

Age in **seconds** of the oldest message on the queue

Estimations of the time in **microseconds** that messages are waiting on the queue for processing.
First value: Calculated from recent activity
Second value: Calculated from longer term activity

Real-time/online monitoring – channels

```
ALTER QMGR MONCHL(MEDIUM) MONACLS(MEDIUM)
ALTER CHANNEL(CLUSTER1.QM1) CHLTYPE(CLUSRCVR) MONCHL(HIGH)
```

- Gives live view of channel throughput

```
DIS CHSTATUS(MQHUB.GATEWAY2) ALL
```

```
AMQ8417: Display Channel Status details.
   CHANNEL(MQHUB.GATEWAY2)                 CHLTYPE(CLUSSDR)
   BATCHES(52)                             BATCHSZ(50)
   BUFSRCVD(55)                            BUFSSENT(1616)
   BYTSRCVD(1748)                          BYTSSENT(1192330)
   CHSTADA(2014-04-08)                     CHSTATI(17.49.03)
   COMPHDR(NONE,NONE)                      COMPMSG(NONE,NONE)
   COMPRATE(0,0)                           COMPTIME(0,0)
   CONNAME(127.0.0.1(1422))                CURLUWID(0107445310001B34)
   CURMSGS(50)                             CURRENT
   CURSEQNO(11823)                         EXITTIME(0,0)
   HBINT(5)                                INDOUBT(YES)
   JOBNAME(00002DA4000047D0)               LOCLADDR(127.0.0.1(53557))
   LONGRTS(999999999)                      LSTLUWID(0107445310001B33)
   LSTMSGDA(2014-04-08)                    LSTMSGTI(17.49.51)
   LSTSEQNO(11773)                         MCASTAT(RUNNING)
   MONCHL(MEDIUM)                          MSGS(1580)
   NETTIME(137538,29555)                   NPMSPEED(FAST)
   RQMNAME(GATEWAY2)                       SHORTRTS(180)
   SSLCERTI( )                             SSLKEYDA( )
   SSLKEYTI( )                             SSLPEER( )
   SSLRKEYS(0)                             STATUS(RUNNING)
   STOPREQ(NO)                             SUBSTATE(RECEIVE)
   XBATCHSZ(20,17)                         XMITQ(SYSTEM.CLUSTER.TRANSMIT.QUEUE)
   XQMSGSA(112)                            XQTIME(545784,3929968)
   RVERSION(07050002)                      RPRODUCT(MQMM)
```

Short/long term calculations of how full your batches are getting, to help you tune BATCHSZ/BATCHINT

Last time a message was sent over the channel

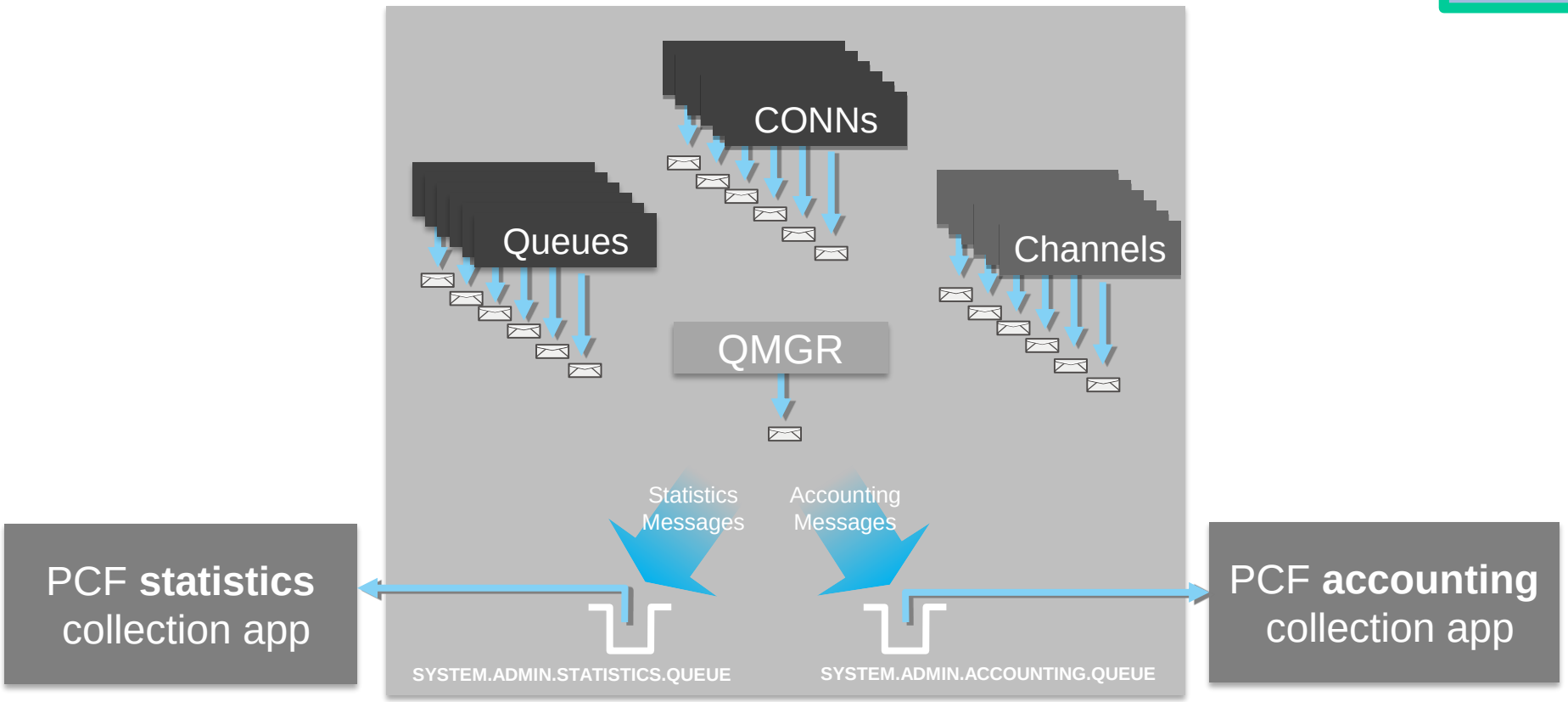
Depth of messages on XMITQ for **this** channel (capped at 999)

Short/long term calculations of how long messages are waiting on the XMITQ for transmission

New options available in V9

Different on z/OS SMF used instead

Accounting and statistics overview



- **Monitoring data sent as a PCF message at a configured interval**
  - Statistics – scoped to a queue / channel / QMGR
  - Accounting – scoped to an individual CONN and queue / QMGR
- **Applications must consume the messages**
  - Enterprise monitoring – Tivoli ITCAM / Tivoli OMEGAMON XE for Messaging
  - Sample applications and SupportPacs, for example MS0P provides MQ Explorer plugins
  - Custom applications – `com.ibm.mq.headers` Java package

New options available in V9

Different on z/OS SMF used instead

Simple practical example using amqsmon

```
ALTER QMGR STATMQI(ON)
Wait a bit, but not the default 30 minutes between stats records
RESET QMGR TYPE(STATISTICS)
amqsmon -m GATEWAY1 -t statistics -a -w 0
```

```
MonitoringType: MQIStatistics
QueueManager: 'GATEWAY1'
IntervalStartDate: '2014-04-09'
IntervalStartTime: '00.00.35'
IntervalEndDate: '2014-04-09'
IntervalEndTime: '00.01.13'
CommandLevel: 700
ConnCount: 35

PutCount: [271, 0]
PutFailCount: 0
Put1Count: [2, 0]
Put1FailCount: 0
PutBytes: [273976, 0]
GetCount: [270, 0]
GetBytes: [269468, 0]
GetFailCount: 19

DurableSubscriptionHighWater: [0, 0, 0, 0]
DurableSubscriptionLowWater: [0, 0, 0, 0]
NonDurableSubscriptionHighWater: [0, 0, 0, 0]
NonDurableSubscriptionLowWater: [0, 0, 0, 0]
PutTopicCount: [0, 0]
PutTopicFailCount: 0
Put1TopicCount: [0, 0]
Put1TopicFailCount: 0
PutTopicBytes: [0, 0]
PublishMsgCount: [0, 0]
PublishMsgBytes: [0, 0]
```

- **Overall QMGR busyness**
- **Simple data format**
  - Multiple values are `[Persistent, NonPersistent]`
- **One message every X seconds**
  - Use amqsmon directly (perl/cron)
- **Low/high water marks for Subscriptions**
  - Grouped by subscription type
- **amqsmon is a sample so you can use it as a base for your own tools**

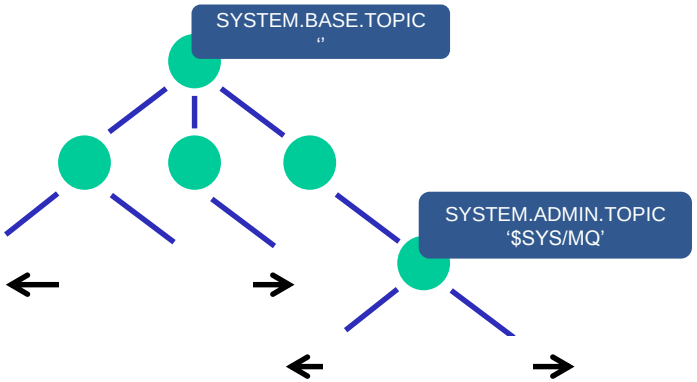
Appliance/V9

Not available on z/OS.

System topics on distributed queue managers

- **Distributed queue manager information is published to a range of system topic strings**
  - ***$SYS/MQ/INFO/QMGR/….***

- **Authorised subscriptions receive their own stream of publications based on the topic string**
  - Administrative subscriptions
    - E.g. For information to be continually sent to defined queues
  - Application subscriptions
    - E.g. To dynamically listen to information as required



- **Unlocks system level information for MQ administrators and DevOps teams**
  - Administrators can grant access to subsets of the data, pertinent to different application teams

Appliance/V9

Not available on z/OS.

System Monitoring

- **Familiar statistics available through subscriptions**
  - Queue manager wide statistics (connects, disconnects, opens, closes, puts, gets, …)
  - Queue level statistics (opens, closes, puts, gets, …)
  - NB: statistics available from system topics are not a 1-1 mapping to those available from system queues
    - E.g no channel statistics, some missing information, some new information, some merged information
  - No support for accounting data

- **Extended to include CPU and disk usage. For example…**
  - Queue manager **CPU time**, **memory usage**
  - **Disk reads/writes**, **disk latency**,

- **Subscribe to meta-topic to learn which classes of statistics are available**
  - $SYS/MQ/INFO/QMGR/QMGR1/**Monitor**/METADATA/CLASSES
  - Then subscribe to specific topics
    - $SYS/MQ/INFO/QMGR/QMGR-NAME/Monitor/class[/instance]/type]
  - See amqsrua sample program

Appliance/V9

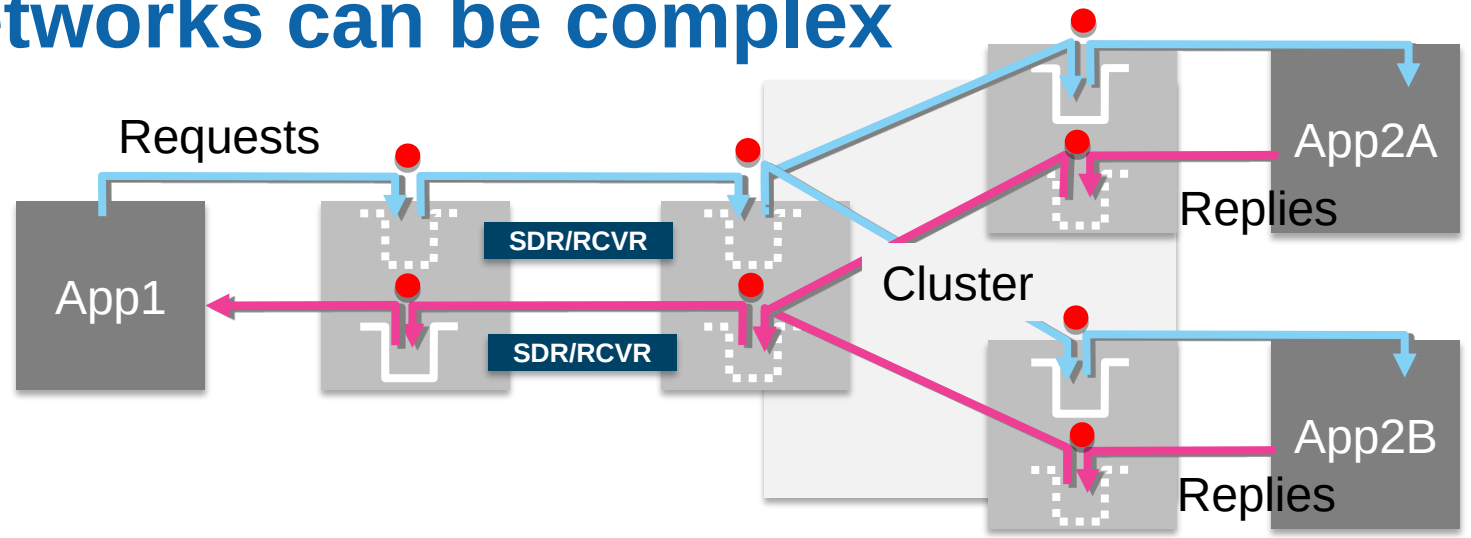
Not available on z/OS

System Monitoring Sample

```
$ amqsrua -m V9000_A
CPU : Platform central processing units
DISK : Platform persistent data stores
STATMQI : API usage statistics
STATQ : API per-queue usage statistics
Enter Class selection
 ==> CPU
SystemSummary : CPU performance - platform wide
QMgrSummary : CPU performance - running queue manager
Enter Type selection
 ==> SystemSummary
Publication received PutDate:20160411 PutTime:10465573
User CPU time percentage 0.01%
System CPU time percentage 1.30%
CPU load - one minute average 8.00
CPU load - five minute average 7.50
CPU load - fifteen minute average 7.30
RAM free percentage 2.02%
RAM total bytes 8192MB
Publication received PutDate:20160411 PutTime:10466573
User CPU time percentage 0.01%
System CPU time percentage 1.30%
…
```

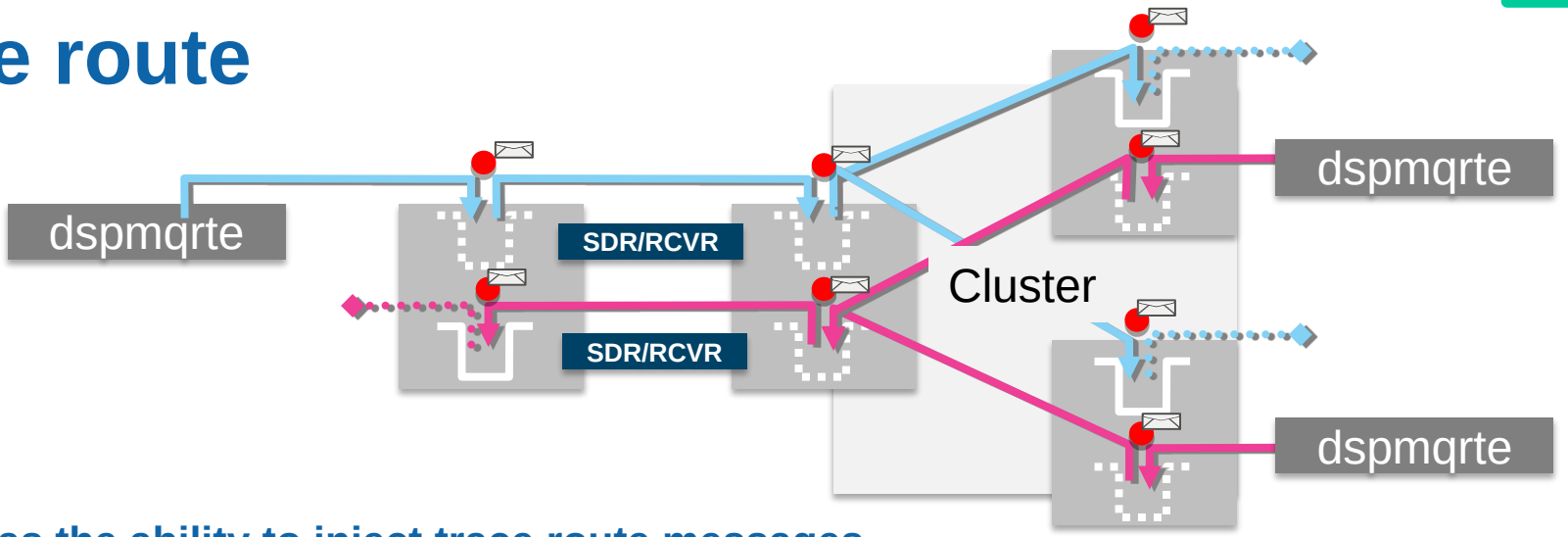
Where are messages going?

MQ networks can be complex



- **At each of the ● dots stuck / mis-sent messages are possible**
  - MQOPENs of the wrong queue / queue manager by apps
  - Full queues
  - Stopped channels
  - Stopped apps
  - Incorrectly configured QREMOTE/QALIAS routing objects
  - Cluster membership problems
- **The standard problem diagnosis approach**
  - Methodically checking channels/queues/DLQs at each point
- **Is there anything to speed up this process?**

Trace route



- **MQ has the ability to inject trace route messages**
  - (Can be) hidden from applications
  - Generate **activity reports** as they pass through, potentially accumulated in the message
- **Tools are available to trace routes using these reports**
  - dspmqrte – command line tool supplied with the product
  - MS0P – Cat 2 SupportPac extension to MQ Explorer
- **Lets you see the path messages *could* have taken**
  - Test connectivity through the MQ network
  - Test cluster workload balancing
- **Can quickly jump you close to the problem**
  - The point your trace message veers off in the wrong direction
  - The point the trail goes cold
- **NB: there is also a related technology: activity recording which generates activity reports from real messages**

What are the apps doing?

New options available in V9

Not available on z/OS

Application activity trace

- **Information on all the MQI operations performed by an application in the order that they were done**

- **Similar infrastructure to accounting & statistics**
  - PCF messages on SYSTEM.ADMIN.TRACE.ACTIVITY.QUEUE
  - Configurable via mqat.ini
    - Can be changed without queue manager restart
    - Configurable detail level can include partial/full message payload
    - Frequency options for tuning
  - Can be enabled on a per-application basis
    - Via MQCONNX flags
    - Via application name

Health warning: Performance impact
http://ow.ly/vA8wB

New options available in V9

Not available on z/OS

Application activity trace

- **Enables scenarios such as**
  - Application audit trail
  - Message duplication
  - Resource usage
    - Which queues or topics are actually being used
  - Problem determination
    - Which queue / queue manager is the application actually opening
  - Application coding standards
    - Does everyone use the MQI in the recommended way
  - And more …

Health warning: Performance impact
http://ow.ly/vA8wB

New options available in V9

Not available on z/OS

Looking at the data with the amqsact sample

```
ALTER QMGR ACTVTRC(ON)          <-  should be tuned via mqat.ini
amqsact -m GATEWAY1 -v
```

```
MQI Operation: 6
  Operation Id: MQXF_PUT
  ApplicationTid: 12451
  OperationDate: '2014-04-09'
  OperationTime: '01:39:48'
  High Res Time: 1397003988665548
  Completion Code: MQCC_OK
  Reason Code: 0
  Hobj: 18225032
  Put Options: 139330
  Msg length: 460
  Recs_present: 0
  Known_dest_count: 1
  Unknown_dest_count: 0
  Invalid_dest_count: 0
  Object_type: MQOT_Q
  Object_name: 'SENDINGAPP.REPLY'
  Object_Q_mgr_name: 'GATEWAY1'
  Resolved_Q_Name: 'SENDINGAPP.REPLY'
  Resolved_Q_mgr: 'GATEWAY1'
  Resolved_local_Q_name: 'SENDINGAPP.REPLY'
  Resolved_local_Q_mgr: 'GATEWAY1'
  Resolved_type: MQOT_Q
  Report Options: 0
  Msg_type: MQMT_DATAGRAM
  Expiry: -1
  Format_name: 'MQHRF2'
  Priority: 4
  Persistence: 0
  Msg_id:
  00000000:  414D 5120 4741 5445 5741 5931 2020 2020  'AMQ GATEWAY1    '
  00000010:  0207 4453 2007 2603                      '..DS .&.        '
  Correl_id:
  00000000:  414D 5120 4741 5445 5741 5931 2020 2020  'AMQ GATEWAY1    '
  00000010:  0207 4453 2007 2203                      '..DS ."."       '
  Reply_to_Q : '                                                  ^D'
  Reply_to_Q_Mgr: '                                                 ^C'
  Coded_char_set_id: 1208
  Encoding: 273
  Put_date: '20140409'
  Put_time: '00394866'
```

As of V9 this is known as display mode

Check the options used for coding standards

Check queue name resolution, to find out why messages are going to the wrong place

Track individual messages and request/reply scenarios with Msg_id and Correl_id

Not available on z/OS

Appliance/V9

Application activity trace – system topics

- **Application activity trace enabled through subscriptions rather than queue manager configuration**

- **Subscribe to topic**
  - E.g. $SYS/MQ/INFO/QMGR/QMGR1/**ActivityTrace**/ApplName/amqsput
  - Filter by application name, channel or connection id

- **When a subscription is created, PCF messages start to flow to the subscriber's queue. When subscription is deleted, messages stop**

- **Much easier to get just the data you want!**

Not available on z/OS

Appliance/V9

Application activity trace – dynamic sample

- Sample provided to demonstrate usage and format output
- Example below specifies application name (-a) so uses dynamic mode
- Dynamic mode subscribes to system topic rather than uses system queue
- Channel name and connection id also supported

```
$ amqsact -m QMGR1 -a amqsput -w 60
Subscribing to the activity trace topic:
  '$SYS/MQ/INFO/QMGR/QMGR1/ActivityTrace/ApplName/amqsput'

MonitoringType: MQI Activity Trace
…
QueueManager: 'QMGR1'
ApplicationName: 'amqsput'
Application Type: MQAT_UNIX
…
==========================================================================
  Tid Date       Time      Operation       CompCode      MQRC  HObj (ObjName)

  001 2016-04-14 09:56:53  MQXF_CONNX      MQCC_OK       0000  -
  001 2016-04-14 09:56:53  MQXF_OPEN       MQCC_OK       0000  2 (QUEUE1)
  001 2016-04-14 09:56:53  MQXF_PUT        MQCC_OK       0000  2 (QUEUE1)
  001 2016-04-14 09:56:53  MQXF_PUT        MQCC_OK       0000  2 (QUEUE1)
```

```
$ amqsput QUEUE1 QMGR1
Sample AMQSPUT0 start
target queue is QUEUE1
Hello
World

Sample AMQSPUT0 end

$
```

How can I look back in time?

What happened to my messages at 2am this morning?

- **Enterprise monitoring solution**
  - DLQ alerts, queue depth alerts, channel status alerts
  - Unresolved running units of work
  - Historical MQ monitoring, accounting and stats data
- **App logs from the time of the problem**
  - Exceptions, MQ error codes, timeouts
- **MQ error logs for all qmgrs that could have been involved**
  - Channel errors
  - Authentication issues
- **??? – what else is there**

Different on z/OS
use CSQ1LOGP

What about the MQ recovery log?

- **For persistent messages inside transactions**
  - MQ logs each operation performed
  - Outside of transactions persistent messages **might** be logged

- **Why can't we use this to**
  - Look back in time to 2am and see what happened?
  - Recover the original payload if the app lost the message?
  - See what happened inside long-running units of work?
  - Provide a list of operations within the failed business transaction?

- **MQ documents how you can… *if***
  - You use the text formatting tool provided with MQ (dmpmqlog)
  - The logging is linear so the historical data is available in the tool
  - You follow the right steps to extract data from running qmgrs
  - You do the work to follow through the logs

Different on z/OS
use CSQ1LOGP

dmpmqlog output is readable, but analysis is tedious

```
LOG RECORD - LSN <0:0:954:44817>
**********

HLG Header: lrecsize 873, version 1, rmid 0, eyecatcher HLRH

LogRecdType . . : AQM Put Message (257)
Eyecatcher  . . : ALRH                  Version . . . . : 1
LogRecdLen  . . : 853                   LogRecdOwnr . . : 256    (AQM)
XTranid . . . . : TranType: XA
   XID: formatID 1463898948, gtrid_length 36, bqual_length 54
        gtrid [000001430B3C84EF0000000100000002734721FAD52A950DDA913D08C5C13719A34E164F2]
        bqual
[000001430B3C84EF0000000100000002734721FAD52A950DDA913D08C5C13719A34E164F2000000010000000000
00000000000000001]
QueueName . . . : Not known
Qid . . . . . : {Hash 2147211283, Counter: 5}
ThisLSN . . . . : <0:0:0:0>
PrevLSN . . . . : <0:0:954:43944>

Version . . . . : 4
MapIndex  . . . : 199
PrevLink.Locn . : 102408                PrevLink.Length : 8
PrevDataLink  . : {High 0, Low 103424}
Data.Locn . . . : 103424                Data.Length . . : 613
Data  . . . . . :
00000:  41 51 52 48 04 00 00 00 FF FF FF FF FF FF FF FF    AQRH....ÿÿÿÿÿÿÿÿ
00016:  00 00 00 00 00 00 00 00 C7 00 00 00 02 00 C0 01    ........Ç.....À.
00032:  00 00 00 00 04 00 01 00 A5 00 00 00 00 00 00 00    ....¥...........
00048:  63 00 00 00 41 4D 51 20 49 49 42 30 31 5F 51 4D    c...AMQ IIB01_QM
00064:  20 20 20 20 D9 26 B0 52 20 08 53 F5 30 30 30 30        Ù&°R .Sõ0000
00080:  30 30 39 39 00 00 00 00 00 00 00 00 00 00 00 00    0099............
00096:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00112:  00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00    ................
00128:  00 00 00 00 FF FF FF FF 00 00 00 00 04 00 00 09    ....ÿÿÿÿ........
00144:  00 00 00 00 E3 ED 04 80 10 FD 67 E4 FF FF FF FF    ....ãí.€.ýgäÿÿÿÿ
00160:  4D 44 20 20 01 00 00 00 00 00 00 00 08 00 00 00    MD  ............
00176:  00 00 00 00 11 01 00 00 B8 04 00 00 4D 51 48 52    ........¸...MQHR
00192:  46 32 20 20 04 00 00 00 01 00 00 00 20 20 20 20    F2  ........
00208:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
00224:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
00240:  20 20 20 20 20 20 20 20 20 20 20 20 49 49 42 30                IIB0
00256:  31 5F 51 4D 20 20 20 20 20 20 20 20 20 20 20 20    1_QM
00272:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
00288:  20 20 20 20 20 20 20 20 20 20 20 20 4D 55 53 52                MUSR
00304:  5F 4D 51 41 44 4D 49 4E 16 01 05 15 00 00 00 64    _MQADMIN.......d
00320:  20 3E AC 57 48 B3 09 B8 71 B0 4C F2 03 00 00 00     >¬WH..¸q.Lò....
00336:  00 00 00 00 00 00 00 0B 20 20 20 20 20 20 20 20    ........
00352:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
00368:  20 20 20 20 20 20 20 20 1C 00 00 00 57 65 62 53            ....WebS
00384:  70 68 65 72 65 20 4D 51 20 43 6C 69 65 6E 74 20    phere MQ Client
00400:  66 6F 72 20 4A 61 76 61 32 30 31 33 31 32 31 39    for Java20131219
00416:  31 34 32 32 33 32 32 32 20 20 20 20 00 00 00 00    14223222    ....
00432:  A4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ¤...............
00448:  52 46 48 20 00 00 00 02 00 00 00 A4 00 00 01 11    RFH .......¤....
00464:  00 00 00 04 B8 4D 51 53 54 52 20 20 00 00 00 00    ....¸MQSTR   ....
00480:  00 00 04 B8 00 00 00 20 3C 6D 63 64 3E 3C 4D 73    ...¸...  <mcd><Ms
00496:  64 3E 6A 6D 73 5F 74 65 78 74 3C 2F 4D 73 64 3E    d>jms_text</Msd>
00512:  3C 2F 6D 63 64 3E 20 20 00 00 00 58 3C 6A 6D 73    </mcd>  ...X<jms
00528:  3E 3C 44 73 74 3E 71 75 65 75 65 3A 2F 2F 2F 51    ><Dst>queue:///Q
00544:  31 3C 2F 44 73 74 3E 3C 54 6D 73 3E 31 33 38 37    1</Dst><Tms>1387
00560:  34 36 32 39 35 32 32 32 36 3C 2F 54 6D 73 3E 3C    462952226</Tms><
00576:  43 69 64 3E 30 30 30 30 30 30 39 39 3C 2F 43 69    Cid>00000099</Ci
00592:  64 3E 3C 44 6C 76 3E 32 3C 2F 44 6C 76 3E 3C 2F    d><Dlv>2</Dlv></
00608:  6A 6D 73 3E 61                                     jms>a
```

Ordered unique IDs for each record (LSN)

A set of documented record types

Transaction information with XIDs, or re-used MQ transaction IDs

MQMD header data at discoverable offsets in the hex of a message Put

The message payload itself

- **Wouldn't it be easier to let the computer do the tedious bit?**

Check out dmpmqlog scraper tool

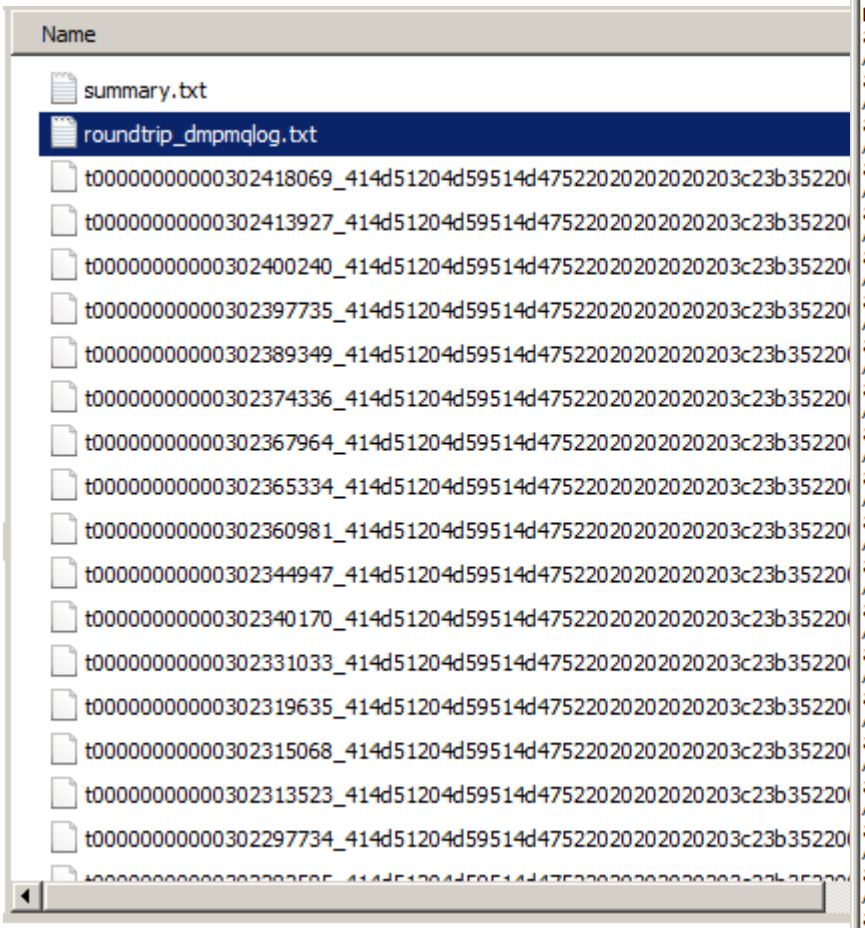
- Takes the tedium out of analysing the output from dmpmqlog
- Created by Peter Broadhurst
- Download from http://www.ibm.com/support/docview.wss?uid=swg21660642

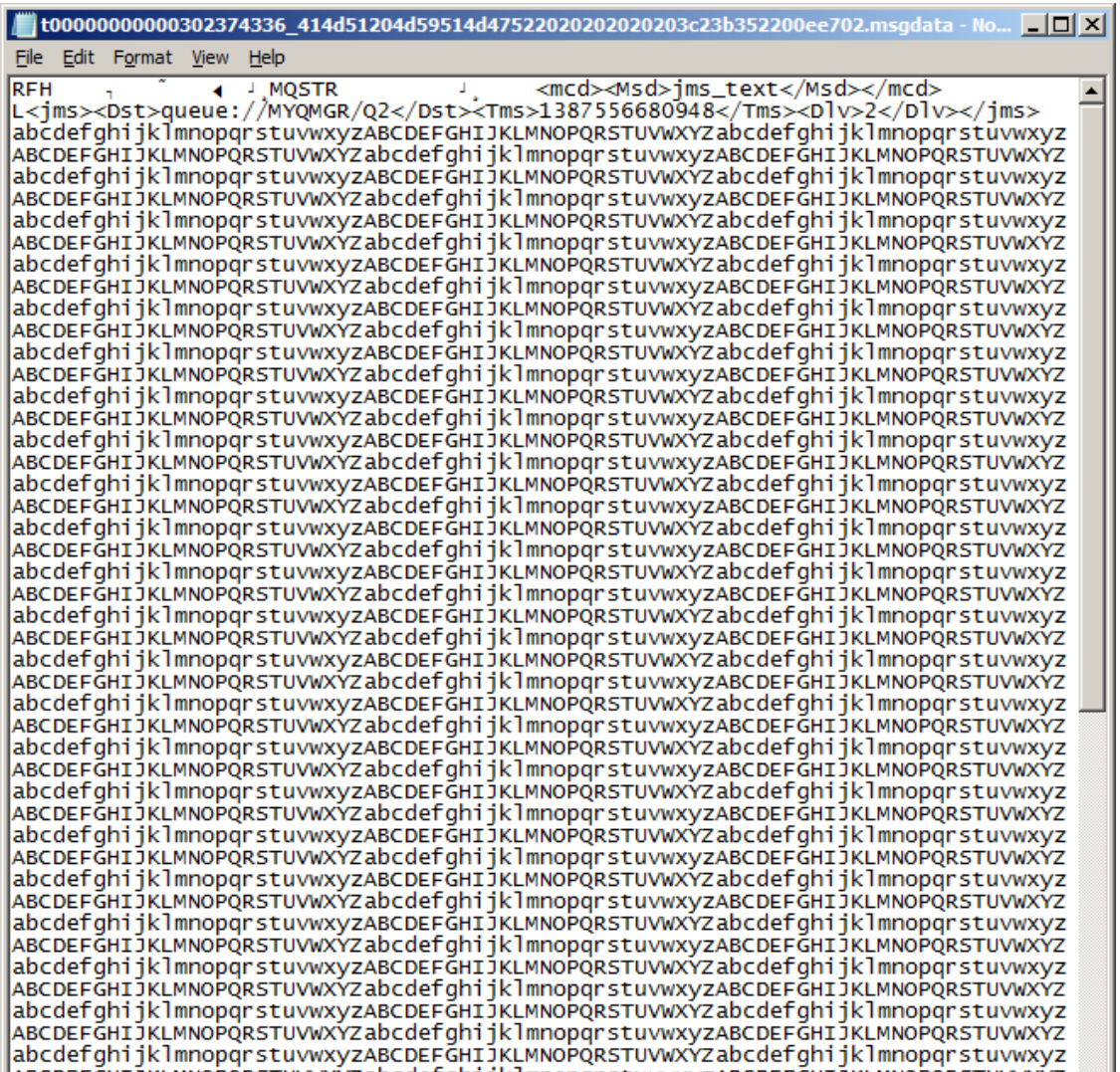
```
java -jar dmpmqlog.scraper-20151201.jar -b little-endian -i dmpmqlog.txt -o .
```

- Generates file per message PUT in the supplied data
- Summary file

dmpmqlog scraper tool output

- Generates file per message PUT in the supplied data.
- Summary file





Summary

- **Lots of tools in your MQ toolbox!**
- **On-line status commands**
  - DISPLAY CONN
  - DISPLAY QSTATUS
  - DISPLAY CHSTATUS
- **Cluster monitoring – AMQSCLM**
- **Off-line statistics and accounting**
  - amqsmon and MS0P to view
- **Tracking**
  - Trace-route
  - Application activity trace
- **MQ recovery logs**
  - dmpmqlog scraper

Questions & Answers