#SHAREorg

SHARE

The Dark Side of Monitoring MQ - SMF Evaluation

Lyn Elkins
elkinsc@us.ibm.com
Session # 11383

SHARE in Anaheim 2012

#SHAREorg

SHARE

The witch trial – MQ is broken!

WITCHCRAFT AT SALEM VILLAGE.

SHARE in Anaheim 2012

#SHAREorg

Agenda

- Introduction to SMF 115
- Introduction to SMF 116 class 3 data
- Summary

#SHAREorg

Objectives

- This session is to introduce the WMQ SMF data and delve a bit into how the ATS team use the SMF data to find transaction and WMQ problems.
- Your buzzword bingo card will remain unmarked.

#SHAREorg

SHARE

Introduction to SMF115

- Statistics records for the Queue Manager
- Enabled via:
  - CSQ6SYSP macro
    - SMFSTAT=YES
  - START TRACE command
    - START TRACE(STAT) DEST(SMF)
- Interval controlled via:
  - CSQ6SYSP macro
    - STATIME=0¦mm
  - SET SYSTEM command
    - SET SYSTEM STATIME(0¦mm)

SHARE in Anaheim 2012

#SHAREorg

SHARE

Introduction to SMF115 cont'd

- Two records cut per SMF interval per queue manager
  - SMF 115 subtype 1
    - Storage Manager and Log Manager
  - SMF 115 subtype 2
    - Buffer Manager, Message Manager, Data Manager, CF Manager, DB2 Manager, Topic Manager, Lock Manager
- Lightweight (subtype 1 < 1K, subtype 2 < 7K)
- Negligible CPU cost
- Recommendations:
  - Always gather and examine this data
  - Useful to store for trend analysis

SHARE in Anaheim 2012

#SHAREorg

SMF115 - continued

- CSQ4SMFD
  - Sample C program shipped with base product to print SMF 115 and 116 records 'dump style' and extract major fields
- MP1B provides several views into the data:
  - MQ1150 – detailed SMF115 report
  - MQCSMF – extracts specific information from SMF115 and 116 in a column format
    - *Particularly useful for building spreadsheets*

SHARE in Anaheim 2012

#SHAREorg

Storage Manager - QSST

```
Storage manager : QSST
  Fixed pools   : Created        48,  Deallocated        49
  Fixed segments: Freed           0,  Expanded          1,  Contracted         1
  Varbl pools   : Created        38,  Deallocated        38
  Varbl segments: Freed        6178,  Expanded        6178,  Contracted         0
  Getmains        48,  Freemains        48,  Non-zero RCs         0
  SOS bits         0,  Contractions         0,  Abends         0
```

- 'SOS bits' (QSSTCRIT) – count of critical short on storage conditions
- 'Contractions' (QSSTCONT) - short on storage was detected and storage contractions had to be done (below-the-bar storage)
- New for V710 – QSSTCN64 and QSSTCR64 – contractions and short on storage for above-the-bar storage (not formatted yet)
- Information not available:
  - High and low watermark use, both below and above the bar
  - Storage use by type (security caching, index, etc.)
  - Storage use in the CHIN by clients and channels

SHARE in Anaheim 2012

#SHAREorg

Storage Manager - Notes

- In addition to the storage manager statistics, review the JES log for the storage use messages
  - If storage use keeps increasing and the free storage goes to less than 100 MB, the queue manager may need to be stopped and restarted to avoid an abend soon. Investigation should take place to determine why storage is not being freed.

```
CSQY220I QML1 Queue manager is using 627 MB of local
storage, 1105 MB are free
```

- Information about the structure storage use can be found in the CF activity reports

SHARE in Anaheim 2012

#SHAREorg

Log Manager – QJST

```
Log manager      : QJST
  Write_Wait          0,  Write_Nowait   3818652,  Write_Force       1663,  WTB        179
  Read_Stor           0,  Read_Active          0,  Read_Archive        0,  TVC          0
  BSDS_Reqs         814,  CIs_Created     750066,  BFWR             103576,  ALR        0
  ALW                 0,  CIs_Offload     914688,  Checkpoints         0
  WUR                 0,  LAMA                 0,  LAMS                0
  Write_Susp     101189,  Write_Reqs       41648,  CI_Writes        758876
  Write_Serl          0,  Write_Thrsh       2381,  Buff_Pagein          0
```

- Note that Checkpoints were 0, but there had been more than 20 during the interval caused by log switches (only counts LOGLOAD checkpoints)
- WTB – is the wait count for unavailable buffers, and the outbuffer value is at the recommended value
- Formula for calculating logging rate:
  - (CI_Writes/ 256) / SMF_interval = MB/min
    (758876 / 256) / 30 = 99 MB/min

SHARE in Anaheim 2012

#SHAREorg

SHARE

Log Manager – QJST cont'd

```
Data compression  : 1
Comp_Req          1,  Comp_fail           0,  Decomp_req          0, Fail         0
Compression:   Before              490,  After              247  49%
Decompression: Before                0,  After                0   0%
Data compression  : 2
Comp_Req          0,  Comp_fail           0,  Decomp_req          0, Fail         0
Compression:   Before                0,  After                0   0%
Decompression: Before                0,  After                0   0%
Data compression  : 3
Comp_Req          0,  Comp_fail           0,  Decomp_req          0, Fail         0
Compression:   Before                0,  After                0   0%
Decompression: Before                0,  After                0   0%
```

- Log compression statistics (COMPLOG=RLE)
  - Message data compressed when writing to log (MQPUT)
  - Three separate compression sections, but only first one used at present

SHARE in Anaheim 2012

#SHAREorg

SHARE

Log Manager – QJST notes

- This is important for customers using a lot of persistent messaging – and those who don't think they are!
- Some of the interesting fields include:
  - Checkpoint
    - Important - only includes when the LOGLOAD has been hit, not when log switching has occurred. May indicate LOGLOAD is too small.
  - Any of the Read_ fields – indicating work is being backed out
  - Wait for buffers, WTB. Increase OUTBUFF.
  - Buff_Pagein. Increase real storage or decrease OUTBUFF.
  - Write force – tasks are suspended until the write completes (commit or out-of-sync)
  - New for V701 – log compression performance
  - CI_Writes – number of 4K CIs written (includes both logs)
- Information not available:
  - Number of log switches / shunts / long-running UOWs

SHARE in Anaheim 2012

#SHAREorg

Message Manager - QMST

```
Message manager : QMST
  MQOPENs    374549,  MQCLOSEs    375694,  MQGETs    5014956, MQPUTs      4564331
  MQPUT1s     89707,  MQINQs       88650,  MQSETs          0, Close_all          0
  MQSUBs          0,  MQSUBRQs         0,  MQCBs           0, MQCBs              0
  MQCTLs          0,  MQSTATs          0,  Publish         0
```

- The message manager reports the number of API requests that have been made
  - NOT the number of successful requests
- Useful for volume tracking – good initial indication of workload change

SHARE in Anaheim 2012

#SHAREorg

Alignment

SHARE in Anaheim 2012

#SHAREorg

SHARE

Buffer Manager - QPST

- Often biggest bang for the buck on performance tuning
- For each bufferpool it reports:
  - The number of pages allocated
  - The 'low' point
  - How the pool is used
  - Short on Storage
- What it doesn't tell you:
  - How many pagesets are used by this pool
  - Number of pages written to/read from each pageset
  - Number of pageset expansions
- It does NO good to increase the bufferpools for shared queues

SHARE in Anaheim 2012

#SHAREorg

SHARE

Buffer Manager – QPST Example

```
> 01 Buffs     15000  Low      0  Now     1844  Getp   351632  Getn   198775
  01 Rio      102140  STW   472341  TPW   260049  WIO    129209  IMW     85105
  01 DWT         137  DMC    81686  STL   276198  STLA        4  SOS       413
```

- The information above was printed by the MQS115 report from MP1B
- While this example is from a stress test, we have seen similar situations in production environments
- If the bufferpool becomes completely exhausted and nothing can be freed, the queue manager will abend with a '**00D70120**' reason code
- There is no indication of pageset expansions, that information can be obtained from the JES log, or DISPLAY USAGE command

```
CSQP017I QML1 CSQPEXT1 EXPANSION STARTED FOR PAGE SET 1
CSQP013I QML1 CSQPEXT1 NEW EXTENT CREATED FOR PAGE
SET 1. NEW EXTENT WILL NOW BE FORMATTED
```

SHARE in Anaheim 2012

#SHAREorg

Buffer Manager – QPST notes

> 01 Buffs	15000	Low	0	Now	1844	Getp	351632	Getn	198775
01 Rio	102140	STW	472341	TPW	260049	WIO	129209	IMW	85105
01 DWT	137	DMC	81686	STL	276198	STLA	4	SOS	413

- Bufferpool churn example from a stress test:
- Note the ‘low’ value of ‘0’ and the SOS value of 413
  - The bufferpool went to sort on storage 413 times in a 5 minute interval
  - There were 102,140 reads from the pagesets
  - There were 129,209 writes to the pagesets
  - The asynchronous write processor was started 137 times (started when number of 'dirty' pages >= 85% total pages)
  - There were 81,686 synchronous writes (number of 'dirty' pages >= 95% total pages)
  - JES log also had repetitions of the following messages

```
CSQP020E QML1 CSQP1RSW Buffer pool 1 is too small
CSQP020E QML1 CSQP3GET Buffer pool 1 is too small
```

SHARE in Anaheim 2012

#SHAREorg

Alignment

SHARE in Anaheim 2012

#SHAREorg

SHARE

DB2 Manager - Q5ST

```
DB2 manager     : Q5ST
  Tasks  : Servers        8,  Active         9,  Conns          0,  Discs         0
           High          14,  Abend          0,  Requeue        0
  Number of deadlock conditions         0
                      Count    Task avg    Task max     DB2 avg     DB2 max  (m/s)
  Reads          :      580           1           2           1           2
  Lists          :      485           4          97           4          97
  SCS Selects    :       30           5          33           5          33
  SCS Inserts    :      212           8          47           8          47
  SCS Updates    :      272           5          49           5          49
  SCS Deletes    :      224           6          25           6          25
  SSK Selects    :       40           0           2           0           2
```

- 'High' represents the high water mark across all requests to the servers.
- 'Task avg' and 'Task max' are the average/maximum elapse time for each request in millisecs. This includes queuing.
- 'DB2 avg' and 'DB2 max' are the average/maximum elapse time for the SQL. This does not include queuing
- SCS / SSK are for shared channels (syncq and keyfile)

SHARE in Anaheim 2012

#SHAREorg

SHARE

DB2 Manager – Q5ST cont'd

```
DB2 manager     : Q5ST
  Tasks  : Servers        8,  Active         9,  Conns          0,  Discs         0
           High           1,  Abend          0,  Requeue        0
  Number of deadlock conditions         0
                      Count    Task avg    Task max     DB2 avg     DB2 max  (m/s)
  Lists          :       62           3          12           3          12
  DB2 MSG Reads  :      300           7         130           7         130
  DB2 MSG Write  :      200          19         926          18         925
  DB2 MSG Delete :      300           8         165           7         165
```

- The above example shows large messages being put to shared queues and offloaded to DB2
  - DB2 MSG Write is for MQPUTs
  - DB2 MSG Read/Delete are for MQGETs

SHARE in Anaheim 2012

#SHAREorg

DB2 Manager – Q5ST Notes

- Only used when in a queue-sharing group
- Is used to report on the queue manager interaction with DB2
- DB2 response time can impact the WMQ response times (MQOPEN) and should be monitored
- Use in conjunction with DB2 performance reports
- High number of Lists – could be due to DISPLAY QLOCAL commands (from monitoring tool perhaps)

#SHAREorg

Alignment

#SHAREorg

SHARE

CF Manager - QEST

```
CF manager      : QEST
  Structure #   0,  Name CSQ_ADMIN    ,  Structure-fulls         0
  Single      168364,  Elapsed time 0000001115116730,  Retries         0
  Multiple      5747,  Elapsed time 0000000257214151,  Retries      1473
  Max entries      708,  Max elements       863
  Structure #   1,  Name APPLS        ,  Structure-fulls         0
  Single      523101,  Elapsed time 000000B7923BCB91,  Retries     11775
  Multiple     14999,  Elapsed time 0000000162517D77,  Retries       280
  Max entries     4997,  Max elements     91409
```

- In the sample above there were no Structure full conditions
- Requests to the CF can be to update a single entry or multiple entries, based on the type of request. They are reported separately in the statistics.
- 'Retries' indicates the number of times a 4K buffer was not sufficient to retrieve the data from the CF and the request had to be retried with a larger (64K) buffer or CF timed-out a request
- 'Elapsed time' is total, in hex (STCK units so divide by decimal 4096 to convert to microseconds; i.e. ignore last 3 hex digits)

SHARE in Anaheim 2012

#SHAREorg

SHARE

CF Manager – QEST Notes

- The CF Manager data
  - Only used when in a queue-sharing group
  - Is used to report on the interaction with the CF structures
  - Should be used in conjunction with the CF Activity Report

SHARE in Anaheim 2012

#SHAREorg

SHARE

Topic Manager – QTST

```
Topic Manager    : QTST
  Subscriptions: Total           4, Durable         0, Expired         0
     API       :  HW mark       29, LW mark        29
     ADMIN     :  HW mark        0, LW mark         0
     PROXY     :  HW mark        0, LW mark         0
  Total msgs to Subscriber queues:   5343115
  Total publication requests:
  -- API:     5343115, ADMIN:          0, PROXY:          0
  Publication fanout information:
  -- HW mark per publish:         32
  -- LW mark per publish:         28
  -- No subscribers:              32
  -- HW mark publish elapse time:          0 m/s
  -- Average Publish elapse time:    5343115 m/s
```

- Details on pub/sub usage
- 'HW mark publish elapse time' and 'Average Publish elapse time' are incorrect – bug in MQ1150 reported!

SHARE in Anaheim 2012

#SHAREorg

SHARE

Topic Manager – QTST Notes

- Subscriptions:
  - API MQSUB
  - ADMIN DEFINE SUB
  - PROXY internal (routing publications through a queue manager)

SHARE in Anaheim 2012

#SHAREorg

SMF 115 subtype 7 – QSRS, Storage Manager Region Summary

- New for V710
- START TRACE(STAT) DEST(SMF) CLASS(1,**2**)

- QSRSLOAL < 16M USER Region alloc value
- QSRSOLOAL previous value of QSRSLOAL
- QSRSELOAL > 16M USER Region alloc value
- QSRSOELOAL previous value of QSRSELOAL
- QSRSGBYTES high water mark for number of usable bytes of above-bar storage
- QSRSAVAL amount of free 31-bit storage
- QSRSAVAL64 amount of free above-bar storage

#SHAREorg

SMF 115 subtype 5 and 6, QSRS and QSGM

- New for V710
- START TRACE(STAT) DEST(SMF) CLASS(1,**3**)
- Also gives QSRS (subtype 7)
- Of most interest to IBM Support

- SMF 115 subtype 5
  - QSPH, Storage Manager Pool Header Statistics
  - One QSPH per storage pool
  - Current and previous size of pool

- SMF 115 subtype 6
  - QSGM,Storage Manager Getmain Statistics
  - One QSGM per module/offset
  - Shows internal getmains by module/offset

#SHAREorg

SHARE

Introduction to SMF116 – Class 3

- Also known as the “New” Accounting records
- Heavyweight – multiple records may be cut for each transaction, and at SMF intervals for long running UoWs
  - Turning this on has been known to swamp an SMF environment
  - But you get marvelous information about what is actually happening
  - Often used in tracking down an application problem and in performance tuning
- START TRACE(ACCTG) DEST(SMF) CLASS(3)
- Recommendation - Even though they are prolific:
  - At least once a month turn on class 3 accounting for one SMF interval
  - Become familiar with the data and with the patterns of WMQ usage

SHARE in Anaheim 2012

#SHAREorg

SHARE

SMF116 Class 3 - Reports

- MP1B provides several views into the data:
  - MQ1160 – prints the SMF116 class 1 report
  - MQ116S – prints the detailed SMF116 class 3 report, including the queue information
  - MQCSMF – extracts specific information from SMF115 and 116 in a column format
    - *Particularly useful for building spreadsheets*

SHARE in Anaheim 2012

#SHAREorg

SMF116 – The Header Information

```
z/OS:Q001  MQ QMGR:QML1  Time: 2010255 13:36:19.73  Jobname:LYNE2054  Userid:MQUSER
          ====> New task record found     <==========
== Thread type.............> RRS BATCH
== Connection name.........> LYNEBTCH
== Operator ID.............> MQUSER
== User ID.................> MQUSER
== Channel name............>
== Chl connection..........>
== Correlator ID...........>
== Correlator ID.....(HEX)> 40404040404040404040
== Context token...........>
== Context token.....(HEX)> 00000000000000000000000000000000
== NID.....................>
== NID...............(HEX)> 40404040404040400000000000000000
== Accounting token........>
== Accounting token..(HEX)> 00000000000000000000000000000000000000000000
== UOW identifier..........>                    Fk™0J€
== UOW identifier....(HEX)> 4040404040404040404040404040c69239F0D1200001
```


#SHAREorg

SMF116 – The Header Information Notes

- The Thread type gives you information about the task, in this case it's a batch process. It may also be mover (for channels), CICS and IMS
- Connection name is the jobname
- The channel name will be present when this is a mover thread
- The correlator ID is not the correlation ID
  - If the SMF data is for a CICS transaction, it will contain the transaction ID. The transaction ID for this record is QPUB and the taskid is 43219:
    - == Correlator ID.....> .®.ÇQPUB.
    - == Correlator ID.....(HEX)> 20AF4B68D8D7E4C20043219C

SHARE in Anaheim 2012

#SHAREorg

SHARE

SMF116 – The Header Information cont'd

```
== Task token : 12-09-2010 17:30:33.73,  3431D3E0,  342E1AE0
== Interval   : START 12-09-2010 17:30:33.73
== Interval   : END   12-09-2010 17:36:19.73
== Number of queue blocks for this task          4
== Other reqs : Count          4,  Avg elapsed        200,  Avg CPU         13
== Latch      : Max number         19,  Max wait   35788780 mics
 > Latch  7,  Total wait        161 mics, Waits           2, Name DMCISTGC
 > Latch 11,  Total wait       6473 mics, Waits           9, Name DMCSEGAL|SSSCONN
 > Latch 12,  Total wait    2483916 mics, Waits         102, Name DMCNMSPC|XMCHASH
 > Latch 15,  Total wait     166693 mics, Waits          55, Name CMXL1    |BMXL1
 > Latch 16,  Total wait      70987 mics, Waits          78, Name BMXL2    |RMCRMST |RLMARQC
 > Latch 19,  Total wait   35788780 mics, Waits        1586, Name BMXL3    |CFXML2  |SRH1_L19
 > Latch 21,  Total wait   18040644 mics, Waits       10680, Name RLMLWRT
 > Latch 24,  Total wait     225667 mics, Waits          53, Name LMXL1
 > Latch 31,  Total wait          0 mics, Waits           2, Name DPSLTCH
 > Latch 32,  Total wait      28816 mics, Waits          45, Name SMCPHB
 > Address of latch for longest wait: 0000000042C37E80
== Commit     : Count        113,  Avg elapsed      53071,  Avg CPU         18
== Log I/O    : Count        461,  Avg elapsed      18574,  Bytes  331798792,
                Forces       445,  Avg elapsed      14012
== Suspend    : Count        113,  Avg elapsed      53051
== Pages      : New        90409,  old      95577
VTASVER 5
== Task token : 12-09-2010 17:30:33.73,  3431D3E0,  342E1AE0
```

SHARE in Anaheim 2012

#SHAREorg

SHARE

SMF116 – The really interesting header Information

- Task token is the task identifying information
- Since this is a long running task, the interval start and end information may be of interest
- The queue blocks gives you the number of queues that have been accessed
- Then there's the latches……

SHARE in Anaheim 2012

#SHAREorg

SMF116 – Latching – The Good, the bad and the …..

```
== Latch     : Max number         19,  Max wait   35788780 mics
 > Latch  7,  Total wait        161 mics, Waits          2, Name DMCISTGC
 > Latch 11,  Total wait       6473 mics, Waits          9, Name DMCSEGAL|SSSCONN
 > Latch 12,  Total wait    2483916 mics, Waits        102, Name DMCNMSPC|XMCHASH
 > Latch 15,  Total wait     166693 mics, Waits         55, Name CMXL1    |BMXL1
 > Latch 16,  Total wait      70987 mics, Waits         78, Name BMXL2    |RMCRMST |RLMARQC
 > Latch 19,  Total wait   35788780 mics, Waits       1586, Name BMXL3    |CFXML2  |SRH1_L19
 > Latch 21,  Total wait   18040644 mics, Waits      10680, Name RLMLWRT
 > Latch 24,  Total wait     225667 mics, Waits         53, Name LMXL1
 > Latch 31,  Total wait          0 mics, Waits          2, Name DPSLTCH
 > Latch 32,  Total wait      28816 mics, Waits         45, Name SMCPHB
 > Address of latch for longest wait: 0000000042C37E80
```

- Latching is performed to serialize requests within the queue manager
- There is always latching going on
  - But there are times when it gets a bit excessive, and needs to be investigated
  - This is one of those times

SHARE in Anaheim 2012

#SHAREorg

SMF116 – Latching – The Good, the bad and the …..Notes

```
== Latch     : Max number         19,  Max wait   35788780 mics
 > Latch  7,  Total wait        161 mics, Waits          2, Name DMCISTGC
 > Latch 11,  Total wait       6473 mics, Waits          9, Name DMCSEGAL|SSSCONN
 > Latch 12,  Total wait    2483916 mics, Waits        102, Name DMCNMSPC|XMCHASH
 > Latch 15,  Total wait     166693 mics, Waits         55, Name CMXL1    |BMXL1
 > Latch 16,  Total wait      70987 mics, Waits         78, Name BMXL2    |RMCRMST |RLMARQC
 > Latch 19,  Total wait   35788780 mics, Waits       1586, Name BMXL3    |CFXML2  |SRH1_L19
 > Latch 21,  Total wait   18040644 mics, Waits      10680, Name RLMLWRT
 > Latch 24,  Total wait     225667 mics, Waits         53, Name LMXL1
 > Latch 31,  Total wait          0 mics, Waits          2, Name DPSLTCH
 > Latch 32,  Total wait      28816 mics, Waits         45, Name SMCPHB
 > Address of latch for longest wait: 0000000042C37E80
```

- The ‘Max number’ is really the latch type that showed the longest wait, in this case latch type 19
- Latch types may be used for multiple purposes
- MP1B has a list of some of the more typical entries, latch 19 is used for serialization to bufferpools
- Latch 21, the second largest wait count, is used when updating log buffers.
- Using these numbers, and looking at the JES message log for the queue manager indicates that during this interval there were numerous log switches and one of the bufferpools expanded
- Further investigation uncovered I/O subsystem issues – the logs and the pagesets were on the same devices for this environment, leading to significant contention

SHARE in Anaheim 2012

#SHAREorg

SHARE

SMF116 – More Header Information

```
 == Commit     : Count        113,  Avg elapsed      53071,  Avg CPU         18
 == Log I/O    : Count        461,  Avg elapsed      18574,  Bytes  331798792,
                 Forces       445,  Avg elapsed      14012
 == Suspend    : Count        113,  Avg elapsed      53051
 == Pages      : New        90409,  Old      95577
WTASVER 5
 == Task token : 12-09-2010 17:30:33.73,  3431D3E0,  342E1AE0
```

- The commit count is useful, especially when working with long running tasks
- The 'Pages' values show how many new and old buffer pages have been used during this interval by this task

SHARE in Anaheim 2012

#SHAREorg

SHARE

SMF116 – Queue Information

```
Open name LYN.LOGQ.Q11                                  Object type:Local Queue
Base name LYN.LOGQ.Q11                                  Base type  :Queue
Queue indexed by NONE
First opened 12-09-2010 17:30:34.17
Last closed  12-09-2010 17:36:19.60
Page set ID            63,  Buffer pool          3
Current opens           0,  Total requests       4157
Generated messages :          0
Persistent messages: GETs          0,  PUTs       4155,  PUT1s          0
Put to waiting getter: PUT          0,  PUT1          0
PUTs: Valid      4155,  Max size     35712,  Min size     17856,  Total bytes    139 MB
-MQ call-          N         ET         CT       Susp       LOGW       PSET Epages   skip expir
 Open   :          1         40         39          0
 Close  :          1          5          5          0
 Put    :       4155      10948        193       9457        915
-Logging: Total-count  Total-elapsed   Force-count Force-elapsed
 MQPUT           326       3.801852          318      2.852607
Maximum depth encountered        385
```

- This is the first queue used by the task
- Detailed information about the queue's use by this task, including:
  - Pageset and bufferpool
  - Number of valid requests
  - Record size range, you an calculate the average size
  - Total elapsed time and cpu time for the requests
  - Maximum depth

SHARE in Anaheim 2012

#SHAREorg

SMF116 – Queue Information

```
Open name LYN.TEST.Q03                                  Object type:Local Queue
Base name LYN.TEST.Q03                                  Base type  :Queue
Queue indexed by NONE
First opened 12-09-2010 17:30:33.73
Last closed  12-09-2010 17:36:19.60
Page set ID             4,  Buffer pool          1
Current opens           0,  Total requests       8518
Generated messages :          0
Persistent messages: GETs       8200,  PUTs          0,  PUT1s          0
Put to waiting getter: PUT          0,  PUT1          0
GETs: Valid      8200,  Max size      7750,  Min size      7750,  Total bytes  63550000
GETs: Dest-S        0,  Dest-G      8515,  Brow-S         0,  Brow-G         0, Su cessful destructive       8200
Time on queue : Max   26.319674,  Min    0.011420, Avg 4294967269.002278
-MQ call-          N         ET         CT       Susp       LOGW       PSET Epages   skip expire
 Open   :          1         71         36         36
 Close  :          1          7          7          0
 Get    :       8515       1608         47       1137          0          0    198      0      0
 Inquire:          1         12          9
-Logging: Total-count  Total-elapsed   Force-count Force-elapsed
 MQGET              2       0.002355            2      0.002355
Maximum depth encountered        299
```

#SHAREorg

SMF116 – Queue Information

- This is the fourth queue used by the task, the ‘get’ queue
- In addition to the information common to all queues, the following should be noted on the GET queues
  - Number of valid gets as compared to the total gets issued
    - The difference means that a number of gets returned no message, often due to a get wait expiring
  - Time on queue
    - In microseconds – though the average often overflows
  - PSET is the average I/O time for a read from a pageset
  - Epages is the number of empty pages there were scanned during a get
  - Skip is the number of pages with messages that were skipped
  - Expire is the number of expired messages that were skipped

#SHAREorg

SMF116 Uses

- Channel usage
- Bufferpool/pageset balancing
  - In a high volume request reply scenario if the two queues are on the same pageset, separating them can improve performance
  - When queues have become concentrated in one resource pool
- Preparation for migration to shared queues
  - Min/Max/Average message size and duration on queue
- Application Performance tuning
  - Proper Indexing
  - Elimination of 'hot spots' – reducing contention
- Problem determination

SHARE in Anaheim 2012

#SHAREorg

SMF116 – What it does not tell you

- Often a consolidated view is needed
  - How many tasks are concurrently using this set of queues?
  - What tasks are related?
    - Can be determined via the queues accessed, but not easily
- Were security calls made during this task?
- No accounting for the IMS Bridge
- Finally, how can the z/OS information and distributed information be consolidated for a complete view?

SHARE in Anaheim 2012

#SHAREorg

SMF116 – A Couple of Gotcha's

- If starting SMF 116 accounting via START TRACE, will only start accounting for subsequent MQOPENs (so long-running tasks like channels may be 'invisible').
  - PM58798 raised to fix this (still open).
- Size of WQSTAT records increased by around 2K between V6 and V701 – so a lot more SMF 116 data written to SMF.
- Long-running tasks have SMF 116 records at STATIME. However, STAT tracing must be started. If not, records only written when task ends.
  - PM46937 fixes this.

SHARE in Anaheim 2012

#SHAREorg

MQCSMF

- Another sample program in MP1B
- Analyzes SMF 115 and 116 records and gives notification of any major problems found. e.g.

```
2000293 VQM2 Buffer pool 3 is too small make larger
2000293 VQM2 Log stats - make OUTBUFF larger.
2000293 VQM2 Archive logs read.
```

SHARE in Anaheim 2012

#SHAREorg

Hunting down the culprit – finding a transaction in the SMF116

- Many times you want to look at the information from a CICS transaction or batch job
  - No way to turn SMF116 class 3 on for just one TX or job
  - Use SORT
    - Remember you will have 2 passes!
      - *First pass to sort out the 'short' records that the SMFDUMP program applies*
      - *Second pass to pull out the records for the transaction/batch job you want*

#SHAREorg

Finding a specific transaction or batch job

- In a group of millions of records, pulling the information for a specific transaction to 'map' it's behavior can be critical in both problem resolution and performance issues
- The SMFDUMP program has few options for getting subsets of the data
- Using a simple sort is a quick solution to dividing up this massive volume into manageable groups

#SHAREorg

SHARE

Finding a transaction

```
//*
//*  THIS GETS RID OF THE 'FIRST AND LAST' SMF RECORDS THAT CAUSE THE
//*  SORT TO COUGH UP BLOOD
//*
//SYSIN  DD *
  OMIT COND=(6,1,CH,LT,X'73')
  SORT FIELDS=(19,4,CH,A)
//
```

```
 //SYSOUT    DD    SYSOUT=*
 //SYSUDUMP  DD    SYSOUT=*
 //*
 //*  SELECT SMF116 BY TRANSACTION
 //*
 //SYSIN   DD *
    SORT FIELDS=(109,4,BI,A)
    INCLUDE COND=(109,4,CH,EQ,C'ABCD')
 /*
```

SHARE in Anaheim 2012

#SHAREorg

SHARE

Finding a Batch job

```
//*
//*  THIS GETS RID OF THE 'FIRST AND LAST' SMF RECORDS THAT CAUSE THE
//*  SORT TO COUGH UP BLOOD
//*
//SYSIN  DD *
  OMIT COND=(6,1
  SORT FIELDS=(1
//
```

```
//SYSUDUMP DD   SYSOUT=*
//*
//*  THIS PULLS THE SMF RECORD FOR A SPECIFIED BATCH JOB
//*
//SYSIN  DD *
   INCLUDE COND=(73,8,CH,EQ,C'ELKINSC2')
   SORT FIELDS=(19,4,CH,A)
/*
```

SHARE in Anaheim 2012

#SHAREorg

SMF116 and Long running tasks

- IF the long running task is started after the Class 3 trace
  - SMF 116 records will be cut at each SMF interval and at task end
- If the task is started before the trace is
  - No records are cut
  - APAR PM58798 has been taken on this

#SHAREorg

Summary

- The SMF data can be used in many ways to find patterns of use, problems with the queue managers, and programming problems.
- There are many other things within the data that are helpful, and more to come with the 7.1 interpretations and print programs. We hope those will be delivered soon!
- Thank you

This was session ???? - The rest of the week ……

#SHAREorg

	Monday	Tuesday	Wednesday	Thursday	Friday
08:00					Free MQ! - MQ Clients and what you can do with them
09:30	Clustering – the easier way to connect your Queue Managers	MQ on z/OS – vivisection	The Dark Side of Monitoring MQ - SMF 115 and 116 record reading and interpretation		
11:00		Diagnosing problems for Message Broker	Lock it down - WebSphere MQ Security	Using IBM WebSphere Application Server and IBM WebSphere MQ Together	Spreading the message – MQ pubsub
12:15	Highly Available Messaging - Rock solid MQ	Putting the web into WebSphere MQ: A look at Web 2.0 technologies	The Doctor is In and Lots of Help with the MQ family - Hands-on Lab		
01:30	WebSphere MQ 101: Introduction to the world's leading messaging provider	What's new in the WebSphere MQ Product Family	Extending IBM WebSphere MQ and WebSphere Message Broker to the Cloud	MQ Performance and Tuning on distributed including internals	
03:00	First steps with WebSphere Message Broker: Application integration for the messy	What's new in Message Broker V8.0	Under the hood of Message Broker on z/OS - WLM, SMF and more	The Do's and Don'ts of z/OS Queue Manager Performance	
04:30	The MQ API for Dummies - the Basics	What the **** is going on in my Queue Manager!?	Diagnosing problems for MQ	Shared Q using Shared Message Data Sets	
06:00			For your eyes only - WebSphere MQ Advanced Message Security	MQ Q-Box - Open Microphone to ask the experts questions	

SHARE in Anaheim 2012

#SHAREorg

The Dark Side of Monitoring MQ - SMF Evaluation

Lyn Elkins
elkinsc@us.ibm.com
Session # 11383

SHARE in Anaheim 2012