PDUG

Gdansk, 2-3 June 2026

WHEN YOU KNOW WHERE TO LOOK, DIAGNOSING DB2 FOR Z/OS PERFORMANCE PROBLEMS IS EASY PEASY

Bart Steegmans

IBM

POLISH DB2 USERS GROUP

Objectives & Agenda

- Objectives
  - Share the experience of diagnosing performance issues to speed up your analysis or prevent issues based on the lessons learned

- Agenda
  - Puzzling performance issue
    - How to investigate an SQL performance issue
  - Challenging slow down cases
    - Complex slowdown cases and z/OS recommendations
  - Sharing some experiences on how to prevent potential problems

Exact same statement yet very different performance

- Reported as “good” and “bad” dynamic SQL
  - Good : Executed using DSNTEP2 - finishes 0.008 second
  - Bad : Executed using Data Studio - uses 30 second and times out
- Statements are identical
  - Good : When the names of the table creator are in lower case
  - Bad : When the creator is in upper case
- Problem started after
  - Increasing the number of partitions using unload / drop / create / reload / runstats / rebind
  - No application changes, no index changes

Documentation to drill down

Documentation collected for a good & bad execution

- Accounting
  - Accounting trace 1,2,3 [7,8,[10]] should always be active

- SQL trace

  -START TRACE(P) CLASS(1,2,3,8) DEST (SMF) IFCID (247,318) AUTHID(xx)
    - Filter on AUTHID to limit the overhead
    - Performance trace class 3 includes all SQL activity PREPARE, OPEN, FETCH,...
    - Include IFCID 247 to include input host variable information
    - Include IFCID 318 for Dynamic Statement Cache (DSC) runtime statistics

Step 1: Observation from Accounting Trace (1)

- Accounting comparison works well when rollup accounting is disabled (ACCUMACC=NO)
  - A quick look at a 'slow' execution
  - Slow execution caused a rollback, so it is typically not part of a rollup record
    - But the good case was rolled up …

```
TIMES/EVENTS  APPL(CL.1)  DB2 (CL.2)     HIGHLIGHTS                    SQL DML     TOTAL
------------  ----------  ----------     --------------------------    -------- --------
ELAPSED TIME   30.484771   30.484768     THREAD TYPE     : DBAT        SELECT          0
                                         TERM.CONDITION  : NORMAL      INSERT          0
CP CPU TIME    12.355929   12.355926     INVOKE REASON   : DEALLOC      ROWS           0
SE CPU TIME    13.600167   13.600167     PARALLELISM     : N/A          IAG1           0
                                         ...                            IAG2           0
SUSPEND TIME    0.000000    0.689257     COMMITS         :          0   UPDATE         0
                                         ROLLBACK        :          1    ROWS          0
NOT ACCOUNT.          N/A   3.839417     ...                            MERGE          0
DB2 ENT/EXIT          N/A          4     MAX WFILE BLKS  :    1907200   DELETE         0
                                                                         ROWS          0

                                                                        DESCRIBE        0
                                                                        DESC.TBL        0
                                                                        PREPARE         1
                                                                        OPEN            1
                                                                        FETCH           0
                                                                         ROWS           0
                                                                        CLOSE           0
```

Lot of CPU and ET, so timed out at client side

High WF activity

Never fetched a row

Step 1: Observation from Accounting Trace (2)

DYNAMIC SQL STMT	TOTAL
REOPTIMIZATION	0
NOT FOUND IN CACHE	0
FOUND IN CACHE	**1**
IMPLICIT PREPARES	0
PREPARES AVOIDED	0
CACHE_LIMIT_EXCEEDED	0
PREP_STMT_PURGED	0
STABILIZED PREPARE	0
CSWL - STMTS PARSED	0
CSWL - LITS REPLACED	0
CSWL - MATCHES FOUND	0
CSWL - DUPLS CREATED	0
...	

RID LIST	TOTAL
USED	0
FAIL-NO STORAGE	0
FAIL-LIMIT EXC.	0
FAIL-NOT CONSTRUCTED	0
INTERRUPTED-NO STORAGE	0
INTERRUPTED-LIMIT EXC.	0
OVERFLOWED-NO STORAGE	0
OVERFLOWED-LIMIT EXC.	**1**
SKIPPED-INDEX KNOWN	0

TOTAL BPOOL ACTIVITY	TOTAL
BPOOL HIT RATIO (%)	100
GETPAGES	**2074918**
BUFFER UPDATES	157411
SYNCHRONOUS WRITE	0
SYNCHRONOUS READ	1908
SEQ. PREFETCH REQS	5529
LIST PREFETCH REQS	0
DYN. PREFETCH REQS	15
PAGES READ ASYNCHR.	7061

Overflow to WF

Step 2: Observations from SQL traces and EXPLAIN

1. Check SQL statement text:

- IFCID 350 (and IFCID 63 (up to 4K)) is part of performance trace class 3 and contains the SQL statement text
  - Besides UPPER vs lower case table creator the statements are identical

2. EXPLAIN both statements:

- Access path of SQL statement with UPPER and lower case table creator is identical

3. Check input host variables:

- IFCID 247 trace
  - The input values are identical (all upper case values)

4. Explain statement with parameter marker vs literal values

- Access path is still identical

Step 3: Drill down SQL execution: Lower case (fast) 1/2

```
PRIMAUTH CONNECT   INSTANCE      END_USER       WS_NAME                        TRANSACT
ORIGAUTH CORRNAME  CONNTYPE      RECORD TIME    DESTNO ACE IFC  DESCRIPTION    DATA
PLANNAME CORRNMBR                TCB CPU TIME              ID
-------- -------- ----------- ----------------- ------ --- --- -------------- ----------------------------------------------------
USER01  SERVER   240706132129 USER01          WORKSTATION0001                w3wp.exe
USER01  w3wp.exe DRDA        09:21:53.09484600 777446  13  64 PREPARE     --> NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
DISTSERV 'BLANK'                    0.02153426                  START          ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F4F34BF2F4F0'
...
                             09:21:53.09487388 777447  13  63 SQL STATEMENT  w3wp.exe
                                    0.02155419                                NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F4F34BF2F4F0'
                                                                              OPTIONS:  X'04' HOST LANG: N/A
                                                                              SQL STMT LENGTH:        553    SQL STATEMENT:
                                                                              SELECT with lowercase creator
                                                                              ...
                                                                              STATEMENT IDENTIFIER:                614319
                                                                              TYPE OF STATEMENT   : DYNAMIC  CCSID:  1208
...
                              09:21:53.09488928 777449  13 350 SQL STATEMENT  w3wp.exe
                                    0.02156359                                NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F4F34BF2F4F0'
        |-------------------------------------------------------------------------------------------------------------------------------
        |OPTIONS      : X'04'          HOST LANG : N/A
        |SQL SEGMENT  : COMPLETE       STMT ID   :                  614319
        |STMT TYPE    : DYNAMIC        CCSID     :                    1208
        |                              SQL LENGTH:        553
        |SQL STATEMENT:
        |SELECT with lowercase creator
        |...
        |-------------------------------------------------------------------------------------------------------------------------------
USER01  w3wp.exe DRDA        09:21:53.09491625 777450  13  58 END SQL     <-- NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
DISTSERV 'BLANK'                    0.02158140                                ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F4F34BF2F4F0'
```

1. Package allocation
2. Prepare start
3. Prepared statement
4. End of prepare (short prepare)

Lower case (fast) 2/2

```
                                09:21:53.09495578 777451   13  65 OPEN          --> w3wp.exe
                                       0.02160475                 CURSOR            NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
...
USER01  SERVER   240706132129 USER01          WORKSTATION0001              w3wp.exe
USER01  w3wp.exe DRDA        09:21:53.09496950 777453  13 247 SQLDA & HOST  NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
DISTSERV 'BLANK'                      0.02161294                VAR TRACING   ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F4F34BF2F4F0'
          |-----------------------------------------------------------------------------------------------------------------------
          |                                              INPUT HOST VARIABLE TRACING
          |LOCATION NAME: DBP2
          |COLLECTION ID: NULLID
          |PROGRAM NAME : SYSSH200
          |STATEMENT NUMBER  :          1       CONSISTENCY TOKEN       : X'5359534C564C3031'
          |LENGTH EACH SQLDA ENTRY:     0       NUMBER ENTRIES IN SQLDA:     2        FORMAT SQLDA          : B'0000'
          |.......................................................................................................................
          | host variable value in hex and char
          |-----------------------------------------------------------------------------------------------------------------------
                                 09:21:53.09499357 777455  13  95 SORT START --> w3wp.exe
                                      0.02162836                              NETWORKID:  NETID001  LUNAME:  LU02        LUWSEQ:     6
                                                                              REQUESTING LOCATION:  ::10.97.34.157
USER01  w3wp.exe DRDA          09:21:53.10356478 777532  13  96 SORT END    <-- NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
DISTSERV 'BLANK'                       0.02232332                              REQUESTING LOCATION:  ::10.97.34.157
                                                                              REQUESTING TIMESTAMP: N/P
                                                                              AR NAME: WORKSTATION0001      PRDID: CLNT/SER V11 R5 M5
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F4F34BF2F4F0'
                                                  ------------------------------------------------------------------------------------
                                                  AREA      :                   156          KEYSZ    :                   92
                                                  WORK      :                     1          RET      :                    0
                                                  ROW DEL   :                     0          PASSES   :                    0
                                                  STMTNO    :                     1          WORKFILES:                    0
                                                  PARTITIONING BY SORT: NO                   SORT IN ADDITION: NO
                                                  PARTIONING OCCURRED : NOT PARTITIONING     TYPE     : ESA
         |SORTL KEY SIZE:                 0
         |
                              09:21:53.10357858 777533  13  59 FETCH        --> w3wp.exe
                                     0.02233713                START          NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
                                                                              ...
                                                                              FETCH SENSITIVITY: UNSPECIFIED
                                                                              FETCH ORIENTATION: NEXT
                              09:21:53.10361385 777537  13  66 CLOSE        --> w3wp.exe
                                     0.02237119                CURSOR         NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F4F34BF2F4F0'
```

Single sort that completes quickly, followed by FETCH and CLOSE

1. OPEN Cursor
2. SQL DA & host var processing
3. Sort start
4. Sort end
5. Fetch start
6. CLOSE Cursor

Step 3: Drill down SQL execution: Upper case (slow) 1/2

```
PRIMAUTH CONNECT   INSTANCE      END_USER       WS_NAME                        TRANSACT
ORIGAUTH CORRNAME  CONNTYPE      RECORD TIME    DESTNO ACE IFC  DESCRIPTION    DATA
PLANNAME CORRNMBR                TCB CPU TIME               ID
-------- -------- ----------- ---------------- ------ --- --- -------------- ----------------------------------------------------
USER01  SERVER   240706132100 USER01          WORKSTATION0001                w3wp.exe
USER01  w3wp.exe DRDA        09:21:02.08174232 484266  11 177 PACKAGE         NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
DISTSERV 'BLANK'                    0.00840690                 ALLOCATION     ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
                                                                              LOCATION         : DBP2
...                                                                              COLLECTION ID    : NULLID
                             09:21:02.08176228 484267  11  64 PREPARE     --> w3wp.exe
                                    0.00842682                   START        NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
...
                             09:21:02.08179027 484268  11  63 SQL STATEMENT  w3wp.exe
                                    0.00845484                                NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
                                                                              OPTIONS:  X'04' HOST LANG: N/A
                                                                              SQL STMT LENGTH:        553    SQL STATEMENT:
                                                                              SELECT with UPPERCASE creator
                                                                              ...
                                                                              STATEMENT IDENTIFIER:                508090
                                                                              TYPE OF STATEMENT   : DYNAMIC  CCSID:  1208

USER01  SERVER   240706132100 USER01          WORKSTATION0001                w3wp.exe
USER01  w3wp.exe DRDA        09:21:02.08180351 484270  11 350 SQL STATEMENT  NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
DISTSERV 'BLANK'                    0.00846620                                ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
        |-------------------------------------------------------------------------------------------------------------------------
        |OPTIONS      : X'04'          HOST LANG : N/A
        |SQL SEGMENT  : COMPLETE       STMT ID   :                  508090
        |STMT TYPE    : DYNAMIC        CCSID     :                    1208
        |                              SQL LENGTH:        553
        |SQL STATEMENT:
        |SELECT with UPPERCASE creator
        |...
        |-------------------------------------------------------------------------------------------------------------------------
                             09:21:02.08182350 484271  11  58 END SQL     <-- w3wp.exe
                                    0.00848620                                NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
```

1. Package allocation
2. Prepare start
3. Prepared statement
4. End of prepare (short prepare)

Upper case (slow) 2/2

```
USER01  SERVER   240706132100 USER01          WORKSTATION0001              w3wp.exe
USER01  w3wp.exe DRDA         09:21:02.08184916 484272  11  65 OPEN       --> NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
DISTSERV 'BLANK'                     0.00851044                 CURSOR        ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
                                                                              ....
                                                                              CURSOR CLOSE COMMIT  : IMPLICIT COMMIT

                              09:21:02.08186825 484273  11 247 SQLDA & HOST   w3wp.exe
                                     0.00852276                 VAR TRACING   NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
      |-------------------------------------------------------------------------------------------------------------------------------
      |                                        INPUT HOST VARIABLE TRACING
      |LOCATION NAME: DBP2
      |COLLECTION ID: NULLID
      |PROGRAM NAME : SYSSH200
      |STATEMENT NUMBER  :          1      CONSISTENCY TOKEN      : X'5359534C564C3031'
      |LENGTH EACH SQLDA ENTRY:     0      NUMBER ENTRIES IN SQLDA:     2       FORMAT SQLDA         : B'0000'
      |.............................................................................................................
      | host variable value in hex and char
      |-------------------------------------------------------------------------------------------------------------------------------
                                                                  4276  11  95 SORT START --> w3wp.exe
                                                                              NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
                                                                              REQUESTING LOCATION:  ::10.97.34.157
                                                                              REQUESTING TIMESTAMP: N/P
                                                                              AR NAME: WORKSTATION0001     PRDID: CLNT/SER V11 R5 M5
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
                                                                              NO DATA
                              09:21:02.08207401 484280  11  95 SORT START --> w3wp.exe
                                     0.00872398                               NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
                                                                              REQUESTING LOCATION:  ::10.97.34.157
                                                                              REQUESTING TIMESTAMP: N/P
                                                                              AR NAME: WORKSTATION0001     PRDID: CLNT/SER V11 R5 M5
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
                                                                              NO DATA
...

USER01  SERVER   240706132100 USER01          WORKSTATION0001              w3wp.exe
USER01  w3wp.exe DRDA         09:21:32.56648006 670164  11 239 OVERFLOW       NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     3
DISTSERV 'BLANK'                    25.96447051                 PACKAGE/DBRM  REQUESTING LOCATION:  ::10.97.34.157
                                                                              REQUESTING TIMESTAMP: N/P
                                                                              AR NAME: WORKSTATION0001     PRDID: CLNT/SER V11 R5 M5
                                                                              ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
...
```

Two sorts and then acctg record (query timeout triggering rollback) 30 seconds later

1. OPEN Cursor
2. SQL DA & host var processing
3. Sort start
4. Sort start
5. Cancel after 30 sec (acctg pkg rec)


PDUG

Difference in Lower case vs Upper case statement

- UPPERCASE creator runtime (slow)

```
USER01  SERVER   240706132100 USER01          WORKSTATION0001              w3wp.exe
USER01  w3wp.exe DRDA        09:21:02.08180351 484270  11 350 SQL STATEMENT  NETWORKID:  NETID001  LUNAME:  LU01      LUWSEQ:     2
DISTSERV 'BLANK'                    0.00846620                               ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F2F84BF2F4F0'
        |-------------------------------------------------------------------------------------------------------------------------
        |OPTIONS      : X'04'          HOST LANG : N/A
        |SQL SEGMENT  : COMPLETE       STMT ID   :                    508090
        |STMT TYPE    : DYNAMIC        CCSID     :                      1208
        |                              SQL LENGTH:        553
        |SQL STATEMENT:
        |SELECT with UPPERCASE creator
        |...
        |-------------------------------------------------------------------------------------------------------------------------
```

- Lowercase creator runtime (fast):

```
                  09:21:53.09488928 777449  13 350 SQL STATEMENT  w3wp.exe
                               0.02156359                          NETWORKID:  NETID001  LUNAME:  LU02      LUWSEQ:     6
                                                                   ACCTKN X'F1F04BF9F74BF3F44BF1F5F74BF6F0F2F4F34BF2F4F0'
        |-------------------------------------------------------------------------------------------------------------------------
        |OPTIONS      : X'04'          HOST LANG : N/A
        |SQL SEGMENT  : COMPLETE       STMT ID   :                    614319
        |STMT TYPE    : DYNAMIC        CCSID     :                      1208
        |                              SQL LENGTH:        553
        |SQL STATEMENT:
        |SELECT with lowercase creator
        |...
        |-------------------------------------------------------------------------------------------------------------------------
```

Different statement ID
These statements are in the DSC
(short prepare, no IFCID 22)

Difference in Lower case vs Upper case statement

Both statements are stored in the dynamic statement cache

- Access path looked at originally was from a ‘fresh’ EXPLAIN
  - We need to look at the access paths **as they are stored in the DSC**
- Use **EXPLAIN STMTCACHE STMTID xxxxx** for each statement to get the access path for each with the access path stored in the DSC
- Use **EXPLAIN STMTCACACHE ALL** to extract all stmts from the DSC and check the DSN_STATEMENT_CACHE_TABLE to verify that there are indeed entries for both stmts with UPPERCASE and lowercase creator
  - If IFCID 318 is active you also get the runtime stats both the SQL stmts in the DSC
    - It should show radically different numbers for both statements in this case

Easy peasy!

- EXPLAIN STMTID xxxx shows a very different access path

	QUERY	QB_PAQB_PLNO_MX	TNAME	MET	AT	MC	ACCESS_OBJ	IX_O	PF	JT	QBLOCK	SORTNJ_UJG	TIMESTAMP
1_	508090	1 0 1 0	T1	0	I	1	PRODUCT.IX_TB1	N	L		SELECT	NNN NNN	2024070421255897
2_	508090	1 0 2 0	T2	1	I	1	PRODUCT.IXI_TB2	N			SELECT	NNN NNN	2024070421255897
3_	508090	1 0 3 0	T3	1	I	1	PRODUCT.IXI_TB3_	N		L	SELECT	NNN NNN	2024070421255897
4_	508090	1 0 4 0	T4	1	I	1	PRODUCT.IXI_TB4_	N		L	SELECT	NNN NNN	2024070421255897
5_	508090	1 0 5 0		3		0	.	N			SELECT	NNN NNN	2024070421255897
6_	614319	1 0 1 0	T2	0	I	1	PRODUCT.IXV_TB2	Y	S		SELECT	NNN NNN	2024070521100878
7_	614319	1 0 2 0	T1	1	I	1	PRODUCT.IX_TB1_P	N			SELECT	NNN NNN	2024070521100878
8_	614319	1 0 3 0	T4	1	I	1	PRODUCT.IXI_TB4_	N		L	SELECT	NNN NNN	2024070521100878
9_	614319	1 0 4 0	T3	1	I	1	PRODUCT.IXI_TB3_	N		L	SELECT	NNN NNN	2024070521100878
10_	614319	1 0 5 0		3		0	.	N			SELECT	NNN NNN	2024070521100878

- EXPLAIN STMTCAHCE ALL shows:
  - The SQL using UPPERCASE creator was cached way earlier than the one with the lowercase creator name

STMT_ID	CACHED_TS
508090	2024-07-04-21.25.58.972914
614319	2024-07-05-21.10.08.785668

Root cause

- Timeline of the changes:
  1. CREATEDTS of SYSTABLES, SYSINDEXES and SYSTABLESPACE indicate the objects (TS, TB, IX) have been created on **2024-07-04-20.59.48** as a result of the table restructuring (adding more parts)
  2. Statement with UPPERCASE creator name (bad performance) entered cache soon after (**2024-07-04-21.25.58.972914**) using ‘default stats’
  3. RUNSTATS was executed at **2024-07-05-09.39.16.323252**

```
UPDATE SYSIBM.SYSTABLES SET
CARDF = 1.847654010000000e+08,
NPAGES = 3054234,
PCTROWCOMP = 100,
AVGROWLEN = 51,
NPAGESF = 3.054234000000000e+06,
SPACEF = 1.432800000000000e+07,
PCTPAGES = 91,
STATSTIME = '2024-07-05-09.39.16.323252'
WHERE CREATOR = ‘PRODUCT'
AND NAME = ‘T1';
```

  4. Statement with lowercase creator entered the DSC at **2024-07-05-21.10.08.785668** using up to date stats -> so picked a better access path

Lessons Learned

- Db2 12 introduced the behavior change of the RUNSTATS utility
  - By default RUNSTATS uses **INVALIDATECACHE NO**
  - As a result the stmt with UPPERCASE creator remained in the DSC and continued to use the ‘bad’ access path
  - Running RUNSTATS with REPORT NO UPDATE NONE continues to invalidate statements in the DSC referencing the object that is being RUNSTATS-ed
- Specify **INVALIDATECACHE YES** if you want to make sure that existing stmts in the DSC that reference the object are being invalidated
- After a RUNSTATS with REPORT NO UPDATE NONE the stmt with the UPPERCASE creator went through a new full prepare and picked up the same (good) access path as the stmt using lowercase creator name

Agenda

- Puzzling performance issue
  - How to investigate an SQL performance issue
- Challenging slow down cases
  - Complex slowdown cases and z/OS recommendations
- Sharing experiences on how to prevent potential problems

Challenges in Diagnosing Application Slowdowns

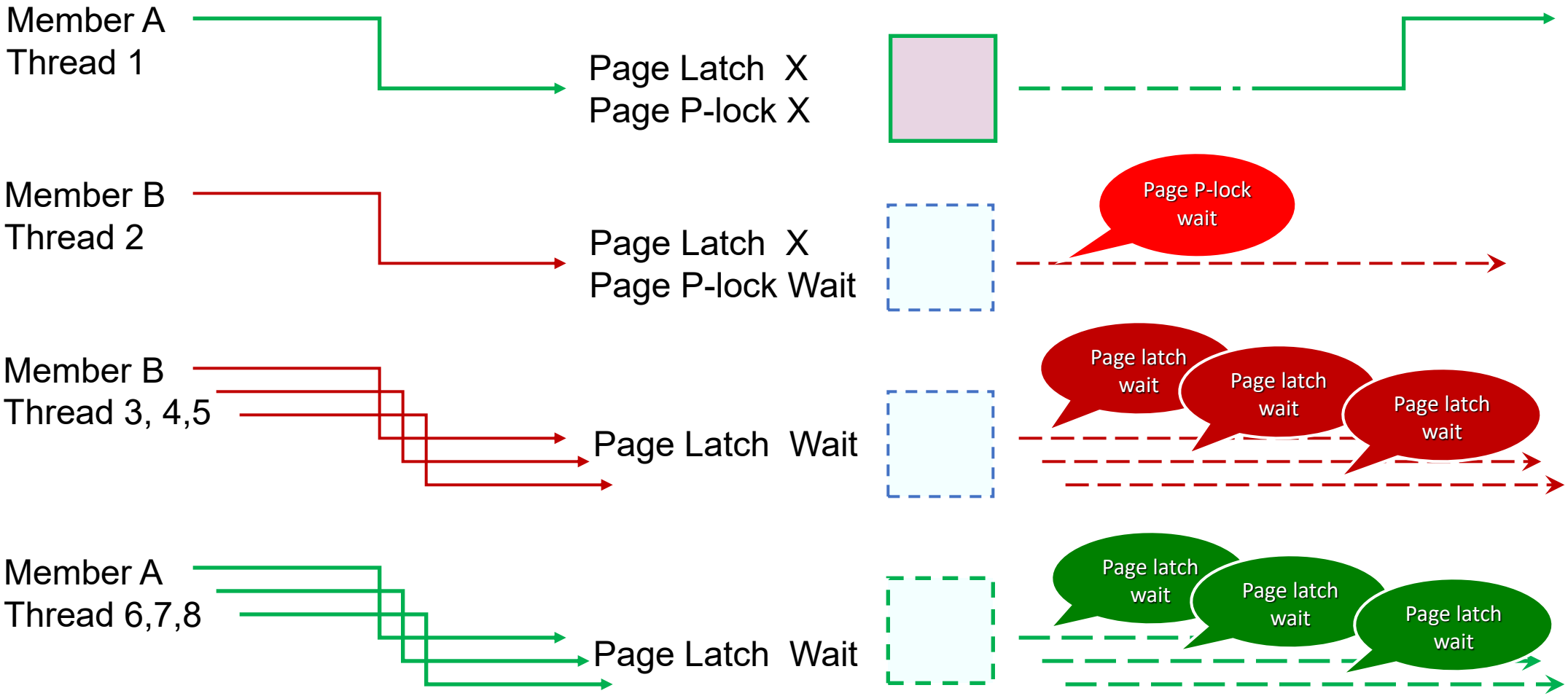
- Critical application slowdown can cause significant impact even a short duration

- Symptoms observed in Db2 in accounting data
  - High class 2 elapsed time
  - High class 3 global lock suspension (typically (page) p-locks)
  - High class 3 Db2 internal latch suspension
  - High class 3 buffer manager page latch suspension
  - [High not accounted time]

- Challenging in a data sharing environment as the cause can be on a different member than where the delays are being reported

- Seemingly widely different symptoms can be related
  1. Interaction between global contention (page P-locks) and page latches
  2. Adding Db2 internal latches to the mix
  3. Adding LPAR reaching capacity to the mix

1. Interaction between Page Latches and Global Contention



(°) Object is GBP-dep and using RLL

Knowing which pages are hot, could help.

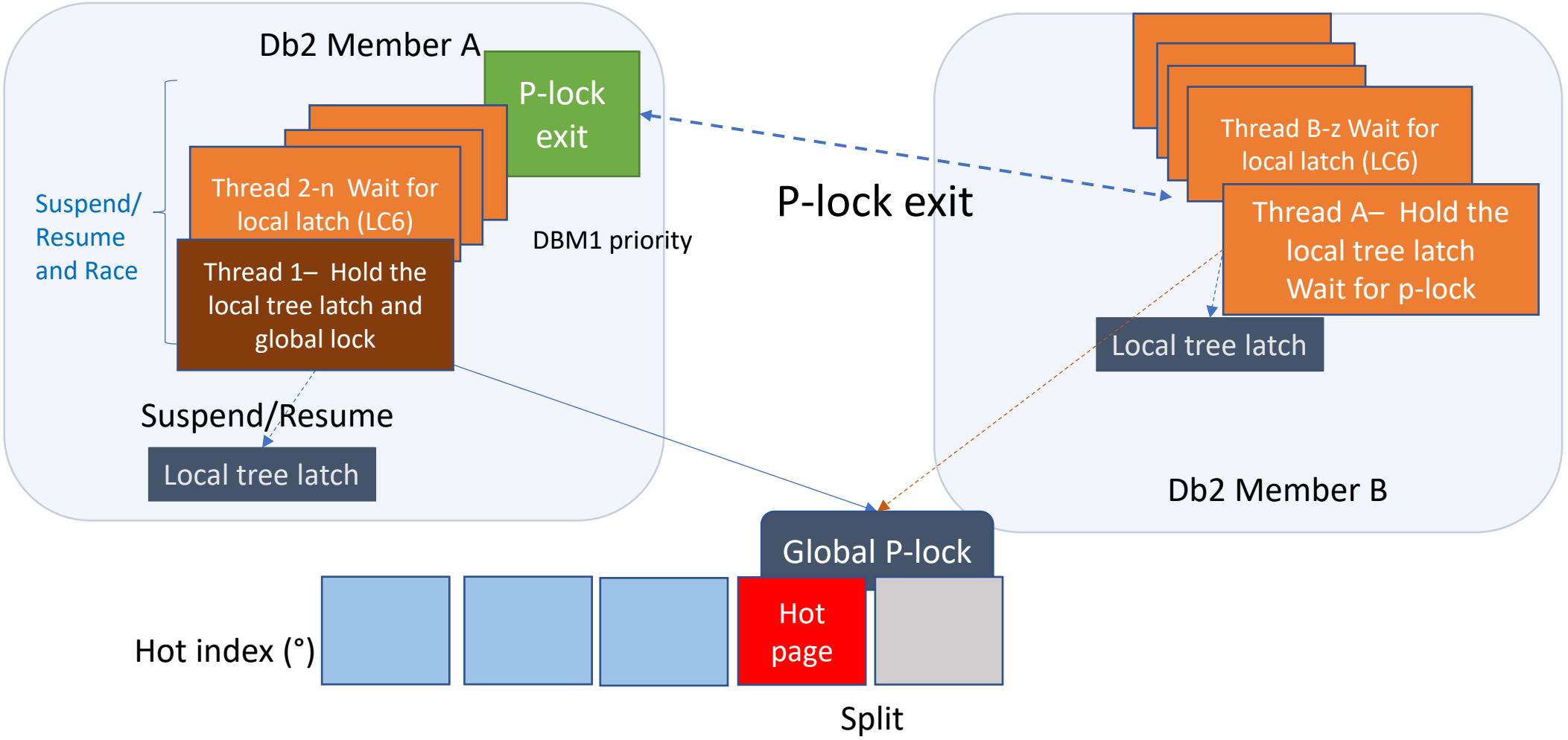
- Db2 13 added “longest waiter info” to the accounting record
  - New DSNDQLLL section in IFCID 3

LONGEST LOCK, LATCH, I/O WAIT	VALUE
ELAPSED TIME	58.599301
WAIT TYPE	DB2 - LATCH
SOURCE ACE	N/P
DBID FOR SYNC/ASYNC I/O	N/P
OBID FOR SYNC/ASYNC I/O	N/P
LOCK HASH	N/P
LOCK NAME	N/P
LOCK RES TYPE	N/A
LATCH CLASS	70
LATCH TOKEN	X'0000005D5254C598'
BEGIN TIME	08:16:39.741141
END TIME	08:17:38.340442

LONGEST PAGE LATCH WAIT	VALUE
ELAPSED TIME	**2.698038**
SOURCE ACE	N/P
DATABASE ID	**384**
PAGESET ID	**33**
PAGE NUMBER	**671633**
PARTITION NUMBER	**N/P**
BEGIN TIME	08:16:25.011905
END TIME	08:16:27.709943

LONGEST SERVICE TASK WAIT	VALUE
ELAPSED TIME	0.003405
SOURCE ACE	N/P
RESOURCE MANAGER ID	3
FUNCTION CODE	73
BEGIN TIME	08:19:22.076850
END TIME	08:19:22.080255

2. Interaction between Db2 Internal Latches and Global Contention



If dispatching priority of the transactions is higher than DBM1 priority, it can cause the delay on p-lock exit

(°) IX is GBP-dep

How to determine where the delays come from

- Detailed **Db2 traces** to understand the flow and the resources involved
  - Not trivial if you don't know when the problem will occur
  - These traces are typically too expensive to be active all the time waiting for the problem to occur
- **Capture dumps** when the problem occurs to see resources involved
  - As we know that Db2 latches, BM page latches, and page p-locks delays of often symptoms of the same problem, use long page p-lock as dump trigger
  - There is a (low overhead) IRLM diagnostic technique to capture a dump of all data sharing members when a thread is actively waiting for a lock for more than x seconds - we picked more than 4 seconds
    - [Page] p-locks can be negotiated - so giving up a p-lock should not take 4 seconds
    - This is a good indicator that something is not performing well
  - [This will require IBM to get involved to analyze the dumps to determine what's causing the delays]

3. LPAR reaching Capacity (Dump Analysis)

- During several hiccups, group wide dumps were captured
- EACH time there was one member that was maxed out (100% CPU busy) and was holding a critical resource delaying the p-lock negotiation
  - Sometimes it was the member requesting the p-lock
  - Sometimes it was the member holding the p-lock
  - Sometimes it was the member that was the XES global lock manager for the lock entry that maps the p-lock
- The ‘blocker’ did not manage to get sufficient CPU resources to complete its work
- As a result the p-lock negotiation could not complete, and
- The work could not continue
  - Remember one ‘blocked’ page p-lock negotiation can prevent a lot of threads from continuing with their work

How to address the challenges

- Don’t run your hardware at 100% busy for an extended period of time
  - Processing capacity should be no more than 92-95% average at peak
    - Especially if there is little non-Db2 ‘displaceable work’
- Detect short periods of 100% busy
  - They often go unnoticed if you only look at RMF data
    - Typically RMF is at 10- 15 min intervals
    - 5-10 sec hiccups are invisible in a 10 min interval
  - Look at SMF 98 or 99 – much more granular
- Defensive mechanisms to help protect you
  - z/OS Boost service Db2 utilizes through
    - –DIS THD(*) SERVICE(WAIT) command or via Db2 internal monitor
      - Good for ‘significant delays’, not designed (aggressive enough) for short hiccups
  - **IRLM PH70343 (3/26)** - Temporarily boost the dispatch priority of the caller of the lock request while in IRLM when CPU usage is over 90%
  - z/OS - Blocked workload support
  - z/OS - WLM policy setup

-DIS THREAD(*) SERVICE(WAIT) SCOPE(GROUP) -1

- Identifies allied agents and distributed DBATs that have been suspended for more than x seconds
  - x = MAX(30, 2x IRLM resource timeout interval) – PH41756 changed 60->30 for first time occurrence, any subsequent check is 60 seconds interval for same thread
- Additional information is displayed to help identify the problem

```
DSNV401I  -DT45 DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  -DT45 ACTIVE THREADS -
           NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
           SERVER RA * 9638 db2jcc_appli ATBNKIO DISTSERV 0244 6212
           V491-LATCH 005D7821CC58 023C HELD BY DB12DBM1 ASID 023C 010.TLPLKNC3
           V490-SUSPENDED 21191-00:50:18.08 DSNB1DCM +00003AE8 UI68078
```

  - DDF threads may return ‘false positives’ – they can be ‘legitimately’ waiting for the next tran or request from the client and may not be stuck in Db2 waiting for a latch
  - PH41756 enhanced info in V491 msg (example on the next slide)

-DIS THREAD(*) SERVICE(WAIT) SCOPE(GROUP) -2

- Automatically driven every 60 seconds by the internal Db2 System Monitor (VMON) and diagnostic information is written out
- Recommendation: save away the output as diagnostics
  - Check DSNV507I, DSNV523I, DSNV491I to determine which threads are being boosted

```
V507-ACTIVE MONITOR, INTERVALS=1235, STG=47%, BOOSTS=1, HEALTH=100
     REGION=1633M, AVAIL=1521M, CUSHION=375M

20.52.49 STC99020 DSNV522I +PB01 DSNVMON - BEGIN DISPLAY OF OLDEST 867
         867 LATCH WAITERS CAUSING A BOOST
20.52.49 STC99020 DSNV523I +PB01 DSNVMON - AGENT 1: 868
         868 NAME ST A REQ ID AUTHID PLAN ASID TOKEN
         868 ---- -- - --- -- ------ ---- ---- -----
         868 PB01 N * 0 020.TLPLKP1C SYSOPR 009E 0
         868 V491-LATCH=005D4B0BE980 LATCH ASID=009E HELD BY JOBNAME=PB01DIST
         868 ASID=009F CORRID=PB0DIA054 TOKEN=170744 ACE=11741820
         868 V490-SUSPENDED 24072-20:46:53.15 DSNKPEXT +00000EA2 UI79153
```

  - Recent PH41756 enhanced DSNV523I / DSNV491I to include latch holder's thread ACE
- Dynamically boosts priority for any latch holder that appears to be stuck - targeted boost via WLM services

z/OS WLM Blocked Workload Support

- Gives small amounts of CPU to stalled dispatchable units of work on the system ready queue
  - z/OS, not specific to Db2 workloads
  - Allows even frozen discretionary jobs to get some small amount of CPU
  - Does not help jobs that are stalled because of WLM resource group capping
  - Does not help zIIPs
- Controlled by two parameters in IEAOPTxx z/OS parmlib
  - **BLWLINTHD** – Threshold wait time before being considered for promotion
    - Default: 20 seconds (too long) – smallest value is 1 second
  - **BLWLTRPCT** – How much of the CPU capacity on the LPAR is to be used to promote blocked workloads : How many units can be promoted at a given time
    - Default: 5 (i.e. 0.5%)
- https://www.ibm.com/support/pages/zos-availability-blocked-workload-support

Blocked Workload Recommendation

- Recommendations
  - All customers should run with this function ENABLED
  - Changing BLWLINTHD to 2-5 seconds may provide better overall system throughput at very high CPU utilisation
  - Regularly review the statistics on this function provided in RMF CPU Activity and Workload Activity reports :

```
BLOCKED WORKLOAD ANALYSIS
OPT PARAMETERS: BLWLTRPCT (%)    0.5  PROMOTE RATE:  DEFINED      13  WAITERS FOR PROMOTE:  AVG 0.010
                BLWLINTHD         20                 USED (%)      4                        PEAK    1


                                        SERVICE CLASS=BATLOW       PERIOD=1 IMPORTANCE=5
---SERVICE---    SERVICE TIME  ---APPL %---   --PROMOTED--   ----STORAGE----
IOC      60439K  CPU 11303.13  CP     294.32  BLK     7.498  AVG     17647.61
CPU     664890K  SRB  176.656  AAPCP    0.00  ENQ     0.000  TOTAL  359251.4
MSO          0   RCT    0.715  IIPCP    2.68  CRM     0.000  SHARED   208.28
SRB      10392K  IIT   73.298                LCK    46.300
TOT     735720K  HST    0.013  AAP       N/A  SUP     0.000  -PAGE-IN RATES-
/SEC    204367   AAP      N/A  IIP     26.62                 SINGLE       0.0
                 IIP  958.208                                BLOCK        0.0
```

z/OS WLM Policy (1/5)

- Discuss with your z/OS team
  - Db2 WLM ‘wish list’
    - https://www.ibm.com/docs/en/db2-for-zos/13?topic=db2-determining-zos-workload-manager-velocity-goals
  - If low priority work is starved while holding a major DB2 internal latch or other vital resource, this can cause an entire DB2 subsystem or data sharing group to stall (as we experienced in this scenario)
  - Remember that online workloads that share DB2 objects should have similar WLM performance goals to prevent interference slowdowns
    - For example, an online DDF enclave-based workload is classified much lower in importance than an online CICS workload
    - If they share any of the DB2 objects, a low running DDF enclave may end up causing the CICS online workload to become stalled
  - Use CPU critical attribute where appropriate
    - For DB2 system AS to protect against stealing of CPU from lower priority work
    - To protect high importance application workloads from being overtaken by lower priority work for a couple of WLM 10 sec adjustment intervals (priority inversion)

z/OS WLM Policy (2/5) : Accounting trace

- An example of very long page latch waits time for DDF work during online day on two members (on one LPAR) of a Db2 data sharing group

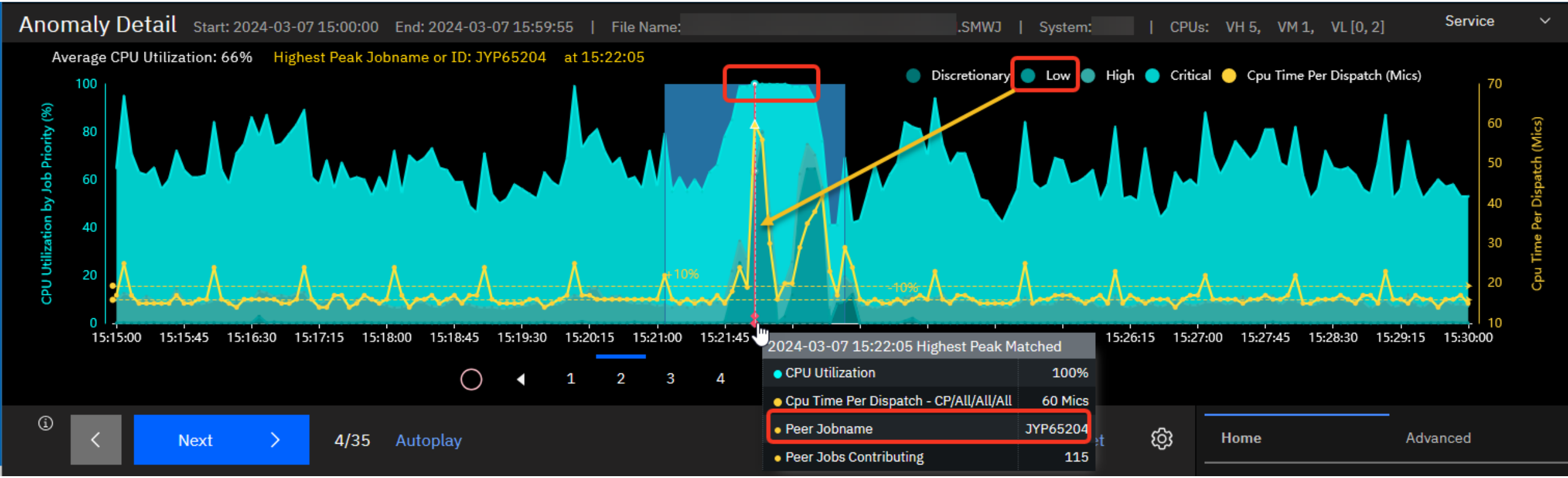
```
---- IDENTIFICATION -------------------------------------------------------------------------------------------
ACCT TSTAMP: 03/07/24 15:22:13.67   PLANNAME: db2jcc_a        WLM SCL: ONLDDF         CICS NET: N/A
BEGIN TIME : 03/07/24 15:22:00.17   PROD TYP: JDBC DRIVER     LUW NET: NET001          CICS LUN: N/A
END TIME   : 03/07/24 15:22:13.67    PROD VER: V4 R24M0       LUW LUN: LU001           CICS INS: N/A
REQUESTER  : ::xx.xx.yyy.zz          CORRNAME: db2jcc_a       LUW INS: DEC23A576B2A
MAINPACK   : SYSLH200                CORRNMBR: ppli           LUW SEQ:        14794    ENDUSER : MYUSERID
PRIMAUTH   : MYUSERID                CONNTYPE: DRDA                                    TRANSACT: db2jcc_application
ORIGAUTH   : MYUSERID                CONNECT : SERVER                                  WSNAME  : aaa.bb.cc.dd

TIMES/EVENTS  APPL(CL.1)  DB2 (CL.2)  CLASS 3 SUSPENSIONS   ELAPSED TIME    EVENTS  TIME/EVENT  HIGHLIGHTS
------------  ----------  ----------  --------------------  ------------  --------  ----------  -------------------------
ELAPSED TIME   13.502976   13.502944  LOCK/LATCH(DB2+IRLM)       0.005154         1    0.005154  THREAD TYPE    : DBAT
 NONNESTED     13.502976   13.502944   IRLM LOCK+LATCH           0.005154         1    0.005154  TERM.CONDITION: NORMAL
 STOR ED PROC   0.000000    0.000000   DB2 LATCH                 0.000000         0         N/C  INVOKE REASON : TYP2 INACT
 UDF            0.000000    0.000000  SYNCHRON. I/O             0.000000         0         N/C  PARALLELISM    : N/A
 TRIGGER        0.000000    0.000000   DATABASE I/O             0.000000         0         N/C  PCA RUP COUNT :          0
                                         READ CACHE HIT         0.000000         0         N/C  RUP AUTONOM.PR:          0
CP CPU TIME     0.001122    0.001093   LOG WRITE I/O             0.000000         0         N/C  AUTONOMOUS PR :          0
 AGENT          0.001122    0.001093  OTHER READ I/O            0.000000         0         N/C  QUANTITY      :          0
  NONNESTED     0.00112     0.001093  OTHER WRTE I/O            0.000000         0         N/C  COMMITS       :          1
                                                                                               SYNCH I/O AVG.:        N/C
SUSPEND TIME    0.000000   11.116851  PAGE LATCH               11.111697         3    3.703899  PROGRAMS      :          1
 AGENT               N/A   11.116851  NOTIFY MSGS               0.000000         0         N/C  MAX CASCADE   :          0
 PAR.TASKS           N/A    0.000000  GLOBAL CONTENTION         0.000000         0         N/C  MAX WFILE BLKS:         64
 STORED PROC    0.000000         N/A  ...                                                       CUR WFILE BLKS:        N/A
 UDF            0.000000         N/A  TOTAL CLASS 3            11.116851         4    2.779213  ZHL READ I/O  :          8

NOT ACCOUNT.         N/A    2.385001
```

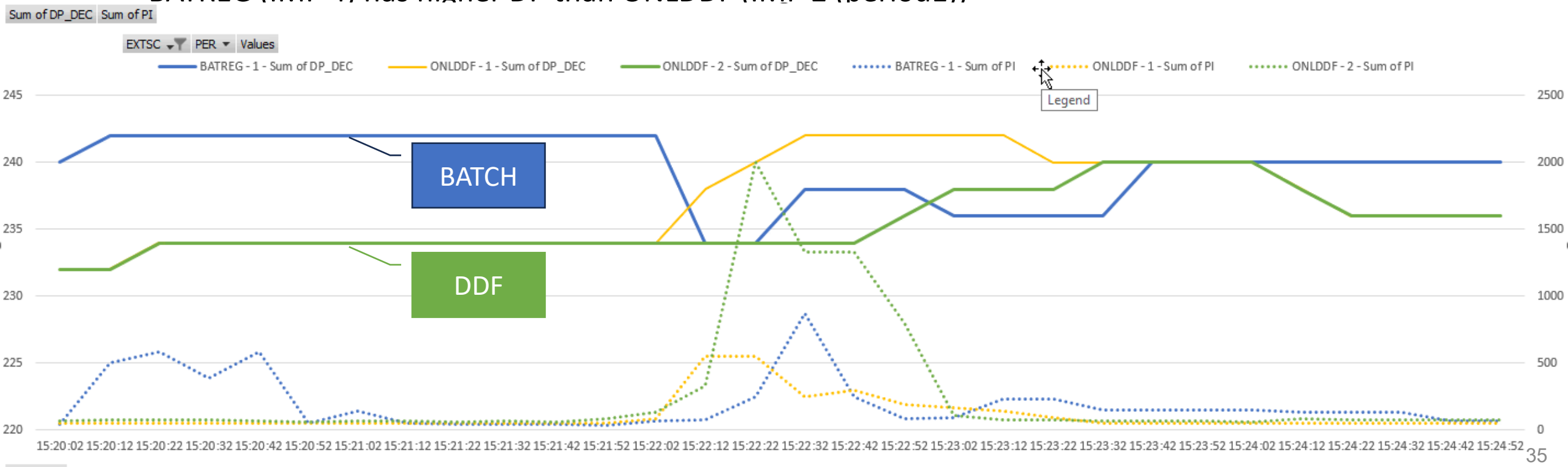
z/OS WLM Policy (3/5) : SMF 98

- SMF 98 (high-frequency throughput statistics) aka correlator data collected at 5 second intervals for the LPAR was available
  - Note the LPAR is 100% around the problem time
  - Seems to be a big spike of ‘low’ priority work taking over the machine



z/OS WLM Policy (4/5) : SMF 99

- Use SMF 99.6 to verify the DP and PI of the batch job(s) and ONLDDF service class the that problem transactions are using
  - DP_DEC – Dispatching Priority in Decimal
  - PI – Performance Index *100 (right axis – dotted lines)
- BATREG (IMP 4) has higher DP than ONLDDF (IMP 2 (period1))



z/OS WLM Policy (5/5) : Recommendation

- ONLDDF is not CPU CRITICAL so lower importance work (like BATREG) can overtake it
  - This is also known as ‘priority inversion’
  - With current definitions it is not possible to use CPU CRITICAL for ONLDDF as the Service Class has more than one period
    - Changed to single period goal and CPU critical and that prevented priority inversion
- Goal of ONLDDF work is too easy to meet so becomes a possible donor when WLM makes policy adjustments
  - RESPONSE TIME 000.00.00.125 FOR 97%
  - But 97.3% finishes in 0.0625 sec (PI 0.5)
    - Goal was tightened and that also helped

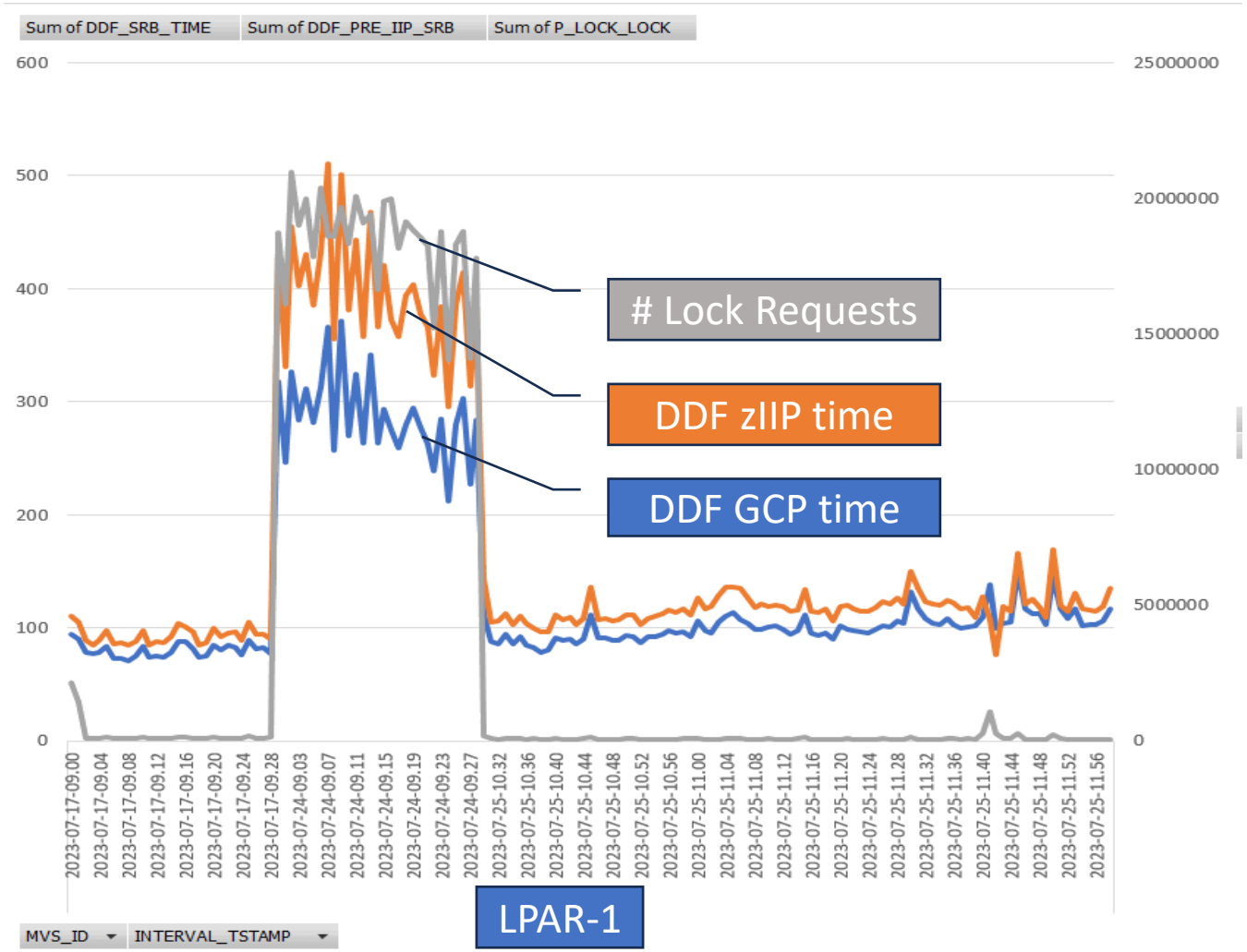
```
----------RESPONSE TIME DISTRIBUTION----------
  -----TIME------  ---# TRANSACTIONS---  ---% TRANSACTIONS---  0    10   20   30   40   50   60   70   80   90
100
   HH.MM.SS.FFFFFF  CUM TOTAL  IN BUCKET  CUM TOTAL  IN BUCKET  |....|....|....|....|....|....|....|....|....|....|
<= 00.00.00.062500      12900K     12900K       97.3       97.3  >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
<= 00.00.00.075000      12957K     57923        97.7        0.4  >
<= 00.00.00.087500      12998K     40974        98.1        0.3  >
<= 00.00.00.100000      13031K     32465        98.3        0.2  >
<= 00.00.00.112500      13054K     23503        98.5        0.2  >
<= 00.00.00.125000      13072K     17363        98.6        0.1  >
...
```

Agenda

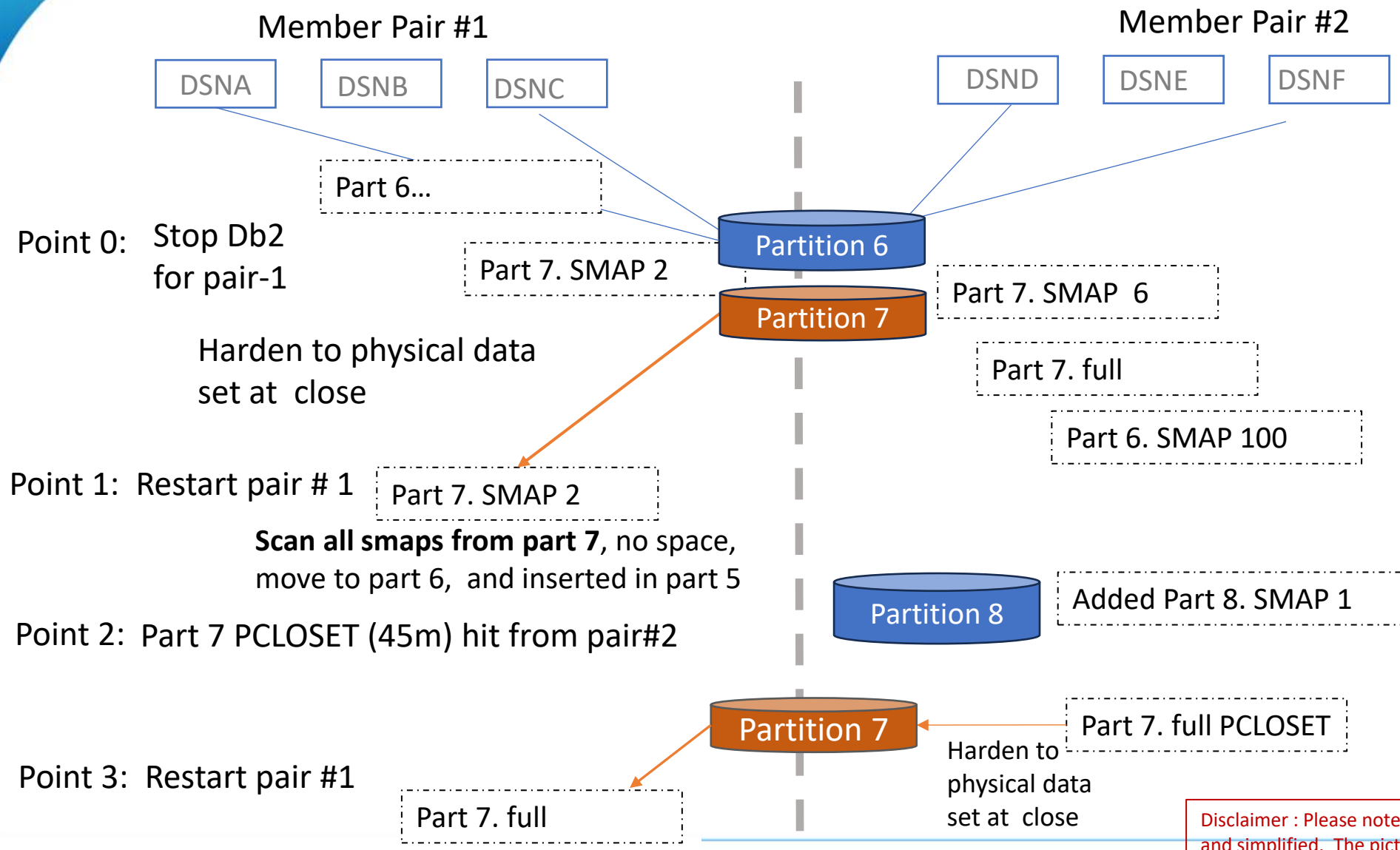
- Puzzling performance issue
  - How to investigate an SQL performance issue
- Challenging slow down cases
  - Complex slowdown cases and z/OS recommendations
- Sharing some experiences on how to prevent potential problems
  - Unbalanced insert between members
  - Top of the hour activity spikes
  - High Db2 not accounted time cause by an inefficient SLIP trap

Alternate members to perform insert

- **Application design**
  - A subset of members perform inserts into the same table (PBG) and alternate the work periodically
- **Symptom**
  - After the switch, observed a large increase in CPU time
    - Due to increased P-lock (space map) requests
      and get pages



Alternate Insert between members



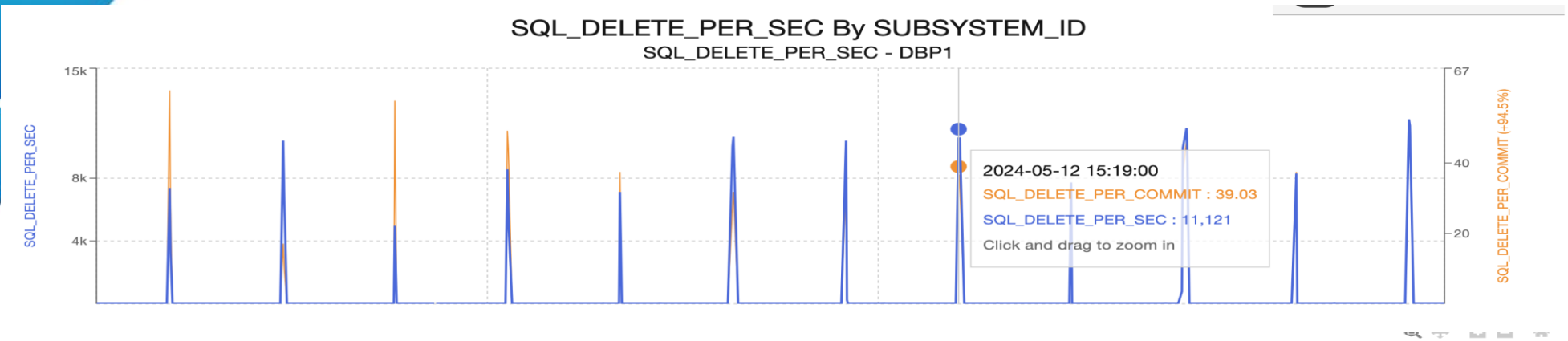
In-memory cache for the last successful inserted position for each partition - Tracked by each member

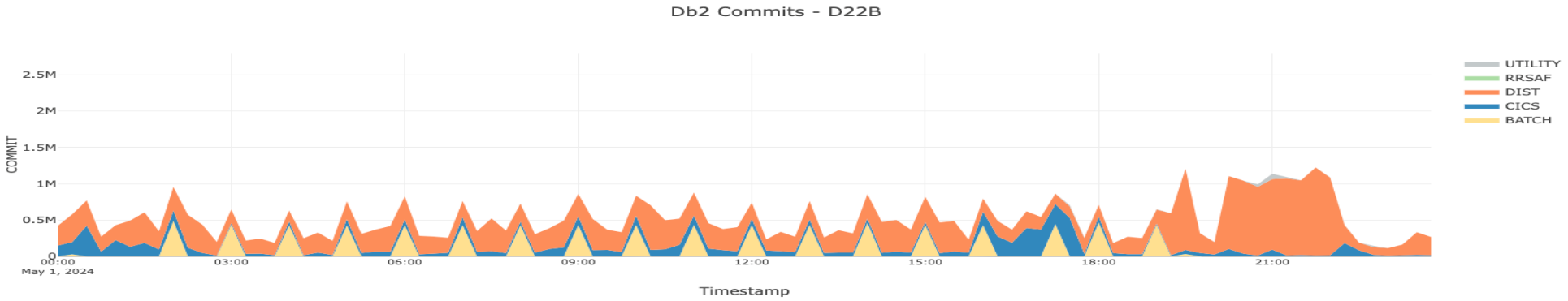
Disclaimer : Please note this picture is created as illustration purpose and simplified. The picture does not contain all the details of Db2 space search algorithm.

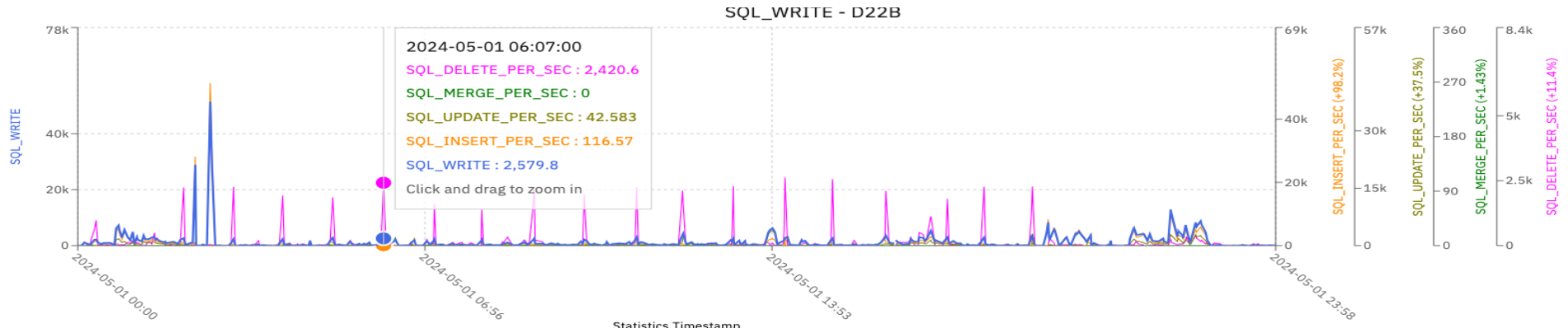
Unbalanced Insert between members

- PH56059 (Aug/2023, HIPER V12 and V13) to address the inefficient space search
  - After the first (long) space map page search detects that a partition is full, update the in-memory information for the last inserted position, even though no row is actually being inserted into that partition (as it is full)
    - Limit the performance issue to the first insert to search through the full set of space map pages.
    - All subsequent transactions inserting into this partition can benefit from the updated in-memory information
    - This applies to **PBG (Partition-by-Growth) TS with MEMBER CLUSTER** attribute. Symptoms can become more severe with larger DSSIZE and APPEND NO.
- Note: To avoid the issue on the very first insert
  - QUIESCE PART LEVEL with WRITE YES against the partitions
    - QUIESCE PART LEVEL from any active members just before the ‘other’ set of Db2 members resume inserts. This will trigger the second set of members to pick up the latest space map information.

Top of the hour activity spikes







- Many customers have a scheduled batch activity at the top of the hour that often triggers a slowdown
- Avoid scheduling too many applications at the top of the hour
- Monitor (e.g. RMF) at the top of hour

Inefficient PER SLIP traps

- **Significant CPU and ET increase with high Db2 not account time**
- Investigation revealed issue started after a PER SLIP was enabled (eg. instruction fetch, storage alteration, successful branch)
- PER (Program Event Recording) SLIP trap
  - PER monitoring by hardware is very efficient
    - Poorly performing PER traps are the typically the result of poor slip trap design
  - Easy check, in any dump of the LPAR, If PSW bit 1 is on, PER slip active
    CPU STATUS:
    PSW=**4**7741001 80000000 00000000 1F645C36
    - x'**4**7' is b'0**1**00 0111' – PER slip active
  - /D SLIP , /D SLIP=xxxx
- Non-PER slips are cheap and safe
  - Uses only software to monitor for error events (capture/suppress a dump)
  - Check for abend (completion) code - COMP=xxx
    - Can be combined with REASON=yyy
  - Check for messages - MSGID=zzz
- If not sure, ask IBM

Summary and Q&A

- Performance diagnostic example and lessons learned
- Recommendations to prevent problems based on challenging experiences

Disclaimer

- IBM’s statements regarding its plans, directions, and intent are subject to change or withdrawal without notice and at IBM’s sole discretion.
- Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.
- The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.
- The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.
- 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.

When you know where to look, Diagnosing Db2 performance problems is easy peasy

Bart Steegmans

bart_Steegmans@be.ibm.com