IP13:WBI-MB for z/OS component performance checklist

Version 1.2

July, 2004

Colin Paice with contributions from Robert Foxon.

Websphere MQ Scenarios department
IBM UK Laboratories
Hursley Park
Winchester
Hampshire
SO21 2JN

Take Note!

Before using this report be sure to read the general information under "Notices".

First Edition, April 2004

This edition name applies to the document *IP13:WBI-MB for z/OS component performance checklist Version 1.2* and to all subsequent versions and modifications until otherwise indicated.

Table of contents

Introduction

This document is a summary of the areas you should check to ensure your system is set up well, especially for performance.

Please send any comments or suggestions for improvements to PAICE@UK.IBM.COM.

Checking out z/OS

This document is a checklist of items you should review to make sure your z/OS system is set up well.

1. Issue D OMVS,L command
    a. WBI-MB can use shared libraries. Check you have enough space by using the D OMVS,L command and check that the difference between the current usage and the limit of the SHRLIBRGNSIZE is more than 11MB.
    b. Also check that the SHRLIBRGNSIZE is not at the maximum value after the broker has started.
2. Issue D OMVS,O
    a. Check MAXASSIZE is at least 1GB
    b. Check MAXCORESIZE is 2GB(2147483647)
3. In some situations using a broker on z/OS 1.4 gives better performance as exception handling is handled a better way.
4. See USS performance tips
   http://www-1.ibm.com/servers/eserver/zseries/zos/unix/bpxa1tun.html
5. If you do not need to audit successful access to a file in the HFS then define a RACF profile see BPX.SAFFASTPATH profile, see UNIX System Services Planning.
6. To get RMF information about HFS, ensure the HFS is added to the ERBRMFxx in parmlib concatenation. For example
   HFSNAME(ADD(HUR.HFS1), ADD(HUR.HFS2))
   Use the USS command *df –P directory_name* to display the dataset name
7. RMF post processing reports should include REPORTS(HFS)
8. Issue command D OMVS,F and locate the HFS name. If the owner is the same as the z/OS image then this is a locally mounted file. If it is different then it is remote, and will require XCF to access the data
9. Ensure the RRS archive logs are large enough. These are optional and you may not use them. See Defining the Log Streams in *MVS Programming Resource Recovery* and the discussion on archive. If the logs are too small then this can impact the throughput when using RRS. The size is defined using LS_SIZE , *see MVS Setting up a Sysplex*.
10. Ensure you have RACF Fastpath enabled. This will cache definition an help speed up access to HFS files.

Checking out Websphere MQ on z/OS

This document is a checklist of items you should review to make sure your Websphere MQ system is set up well.

1. Log size – a 1000 cylinders is typical
2. Logs placed on low activity devices
3. Number of active logs – at least 4 – more if archiving to tape, and you get tape problems which prevent archiving
4. Buffer pools have been set to large, for example 100000, – and tuned down if necessary
5. The default system parameter IDBACK is too small for production work. Make this large, for example 32000.
6. SupportPacs down loaded
    a. MP16 performance and tuning
    b. MP1B accounting and statistics
    c. MO71 MQMON monitoring WMQ from an MQ client
7. Batch adapter modules loaded into the LPA if DB2 stored procedures used
8. Channel exit code loaded into LLA if frequent channel stats
9. Cluster workload exit re-entrant
10. //SYSTCPIP specified in the CHINIT JCL

The following are not performance related

11. All queue managers to have unique port
12. Page sets are backed up daily, and these results checked

Checking out WBIMB on z/OS

This section is a checklist of items you should review to make sure your WBI Message broker system is set up well.

1. For V5 ensure you have the DB2 library DSNLOD2 in your steplib. DB2 APAR PQ74296 provides the XPLINK support required for WBI V5.
2. For V5 ensure you have APAR PQ71096 applied to WMQ.
3. The executables in lil, lib, and bin subdirectories have the **Shared address space = YES** attribute set. From a subdirectory use the **extattr *** command to display this, and set it with **extattr +s ***
4. The **shared library** extended attribute enables multiple processes to share subroutines in object libraries more efficiently. To find out if the shared library extended attribute has been set, use the ls -E command. Set it using **extattr +l *
5. Check the Shared address space attribute forJava
6. WLM definitions in the *classified rules* under both OMVS and STC.
7. Check trace is turned off, check the files in the /log subdirectory are not increasing in size, and the file time stamps are not changing
8. Check the ODBC trace is not set. Check APPLTRACE=0 is set in the DSNAOINI file. The file in output/traceodbc should not exist or it should not be increasing in size.
9. Region size set to 0M and IEFUSI exit is not used to reduce the region size.

The following are not performance related

10. Ensure the broker started task userid has a OMVS UID, and home directory. Use the RACF command *LU xxx OMVS NORACF*
11. Ensure the broker started task userid has a OMVS GroupID.
    - Use the RACF command *LU userid* and note the default group
    - Use the RACF command *LG group OMVS NORACF* to display the group ID
12. You have enough space in the started task userid's home directory, use the **df –P .** from the directory
13. You have enough space in the broker's component directory use the **df –P .** from broker directory. If you have turned on trace then it may use a lot of space, and reduce the amount of free space.

Check out the following documents on designing your WBI flows

1. Design and Implementation Considerations for IBM WebSphere MQ Integrator Message Flows
   (http://www7b.boulder.ibm.com/wsdd/library/techarticles/0211_dunn/dunn.html)
2. Tuning a WebSphere MQ Integrator Broker
   (http://www7b.software.ibm.com/wsdd/library/techarticles/0301_dunn/dunn.html)
3. How to Estimate Message Throughput for an IBM WebSphere MQ Integrator Version 2.1 Message Flow
   (http://www7b.software.ibm.com/wsdd/library/techarticles/0208_dunn/dunn.html)

Checking out DB2 for WBIMB on z/OS

This section is a checklist of items you should review to make sure your DB2 is set up well for WBI-MB. This document assumes that your DB2 is set up for general performance.

1. Check that dynamic caching is on. You can see this from DB2 accounting records. See ***Application programming defaults panel 2: DSNTIP4*** in *DB2 Universal Database for OS/390 and z/OS Installation guide* (**GC26-9936)** or similar manuals. The Dynamic SQL Cache is stored in the EDM pool so if dynamic caching is switched on in an existing DB2 subsystem the EDM pool needs to be increased in size. As well as monitoring this at application level it should also be monitored at system level using the DB2 PM statistics report.
2. Check that the ODBC plan has been bound with KEEPDYNAMIC(YES) when using WBIMB V5.
3. For WBI V5 check that DB2 APAR PQ74296 is applied.
4. Stored procedures called by MQSI should run under WLM if they are to be fully globally co-ordinated. This is because WLM provides a 2 phase commit for non-SQl resources using RRSAF.
5. Any WLM stored procedure needs to be linked with the RRS adapter DSNRLI.
6. Make sure that you run IRLM with the lock information being held in local storage; use the startup parameter PC=YES.
   There is more information about this in the DB2 Redbook "DB2 UDB for OS/390 Version 6 Performance Topics", SG24-5351.
7. Check APAR II16693 and put
   DESCRIBEPARAM=0
   DESCRIBECALL=0
   in dsnaoini
8. Check PQ46261/UQ56466 is applied.
9. Check stored procedures are linked with the correct stub. If you use the wrong stub then you will get a lot of loading DSN3ID00, DSNHDECP,DSNACAB and DSNACAF
10. Isolate LOB tablespaces into a separate bufferpool and set DWQT=VDWQT=0 e.g. ALT BUFFERPOOL(BP32K1) VDWQT(0) DWQT(0).

The following items were contributed by Robert Foxon

DB2 checklist

In the following section are useful pointers for things to check. **T**hey are fairly general and may not be applicable to your system.

Apportionment of CPU Costs.

All CPU costs incurred by DB2 are charged to the application and not to the DB2 started task address spaces. So from outside of DB2 the CPU used by an application includes the processing that it did and any processing by DB2.

DB2 Accounting trace records are able to distinguish between CPU accrued by the application and CPU accrued by DB2 (sometimes known as class 1 and class 2 times).

The only exception to point 1 above is that any asynchronous tasks such as sequential prefetch, normal writing and log off-loads have their CPU cost charged to the DB2 started task.

.

DB2 Work Files.

DB2 uses work files for sorts and visualising results sets from the OPEN CURSOR command. The work files are held in table spaces in database DSNBDB07. It is important that once created, rows are not externalised to DASD. For this reason it is highly recommended that table spaces in DSNDB07 have their own buffer pool that can be tuned to give maximum residency. The DSNDB07 buffer pool should be sized to give a hit rate of around 95%. Note that the calculation for this is different from the calculation for the hit rate for other buffer pools - (Getpage Requests/2 - Sum of all Reads)/Getpage Requests/2 * 100. Set VPSEQT to 95%, DWQT to 85% and VDWQT to 50%. Unlike all other pages, DSNDB07 pages are not written to DASD by checkpoints so setting DWQT and VDWQT high allows a good page residency in the buffer pools.

This list captures the initial areas you need to investigate when looking at performance problems. All too often they are reported as not going fast enough, and other such vague descriptions.

Unfortunately, there is no magic ‘flowchart’ as often you need to use a DB2 monitor (DB2PM etc.) together with accounting traces to try and see where the problem lies.

There are however, a few basics that ought to be checked to ensure that everything is running smoothly before focusing in on a particular area, unless it is just one transaction, one batch program, one type of utility or a particular object that is reported.

The following offers a guide to areas to inspect if you cannot ‘tie down’ the problem specifically.

Bufferpools

DB2 bufferpools play a crucial role in the performance of the sub-system. There are many things to check but they can be itemized as follows:

1) Distribute DB2 objects distributed across the bufferpools
2) Only DB2 catalog and Directory objects in Buffer Pool 0
3) Allocate DSNDB07 workfile tablespaces to their own bufferpool
4) Separate tablespaces and indexspaces (at least) and then consider by type of access (random/sequential)
5) Consider fixing high referenced objects e.g., lookup and code tables in a bufferpool on their own and ‘read them in’ at the ‘start of the day’.
6) Isolate objects that support vendor/ISV tools into their own bufferpool.
7) Are there too many / too few bufferpools ?

a) Do not exceed 0.8 GB for total virtual pool storage.
b) Make sure a 32K bufferpool is allocated - it can be used for long ‘ rows’ built whilst processing objects to form the final result set for a query as well as for objects with row lengths over 4K.
c) Keep a sense of proportion - too many bufferpools can have an adverse effect - you need to ensure that the minimum size of the bufferpool is not so small that decisions DB2 takes over pre-fetch are affected - view differ but allocate at least 1000 pages and an often used rule of thumb is 1401 pages
d) Monitor page residency to help determine if the bufferpool is too large or too small - set a limit and monitor then adjust

8) Have the various bufferpool thresholds been set?
a) Avoid hitting the critical thresholds set at 95% and above
b) Ensure DWQT and VDWQT are set - don’t leave at defaults. VDQWT should be lower than DWQT and should be reduced from the default to somewhere between 1 and 10% initially.
c) If pages written per write I/O is less than 10 set VDWQT to 0%. This will force DB2 to write 32 pages as soon as 40 ‘dirty’ pages are accumulated (for a 4K buffer pool).
d) For DSNDB07 bufferpool set DQWT/VDQWT very high 90/95% due to the way DB2 makes use of workfiles - likewise you can afford to set VPSEQT high as well.
e) Be aware that if your write I/O is slow and DWQT is at 95% then you will hit the sequential prefetch threshold at 95%. (SPTH). I would say no more than 85%. There is a small amount of random access in the work file buffer pool (for sparse indexes amongst other things) but generally this is so small that VPSEQT of 100% is usually OK. Lower thresholds can be advantageous if certain limits are being hit regularly.
f) Consider altering the VPSEQT threshold upwards overnight to reflect the greater bias towards sequential processing in a ‘typical’ batch environment.
g) Make use of the other thresholds to control parallelism and pre-fetch if you experience parallelism fallback e.g. VPPSEQT

9) Hiperpools and data spaces
a) Consider these to get over the 1.6 GB limit for VPs in DBM1 for HPs and as a way moving VPs out of DBM1 for data spaces.
b) Hiper pools are also useful where there are a large number of DB2 subsystems on one LPAR. I have just completed a buffer pool review for a customer who had ten susbsystems on one LPAR. Reducing the virtual pool sizes and using hiper pools allowed central storage to be more fairly shared, reducing paging.
c) Ensure appropriate level of DB2 and operating system for data space use - 64 bit addressing is a practical prerequisite.
d) Consider these to assist in avoiding I/O. Look for a VP:HP ratio to be at least 1:2 and possibly 1:4. But monitor use, they can only be used for pages that have not been updated and if less that 10% of the HP is ever used you might usefully use the expanded storage allocated elsewhere.

10) Memory use

a) In addition to looking at bufferpools the following areas, within the DBM1 address space should be reviewed:
b) Check for sub-system paging - measure the pageins for I/O against total requests and aim for 5% or fewer. Also keep a watch with the help of the MVS sysprogs to ensure that the allocation of bufferpools can be supported by adequate central storage.
c) Monitor the use of sort pool - a balance has to be struck - you will probably never get a sort pool that copes with all sorts but sizing to ensure that it copes with many will take stress of DSNDB07 etc. - again remembering that it could be too large and is allocated per user
d) RID pool - monitor this for failures - this will cause fallback to tablespace scans and a consequent adverse effect on runtimes. Look a explain for queries experiencing failures to possibly change access path if tables involved are very large by rewriting, extra indexes etc.
e) On busy systems set the VSTOR RMF report on the DBM1 address space and monitor the storage buffer available. This will avoid any unexpected 878 or 80A errors, these will eventually bring DB2 down.
f) RID failures may be caused by hybrid joins determined when the plan or package was bound with relatively small amount of data, Runstats and rebind will often cure.
g) Accounting reports will report RID failures for individual plans and packages making the process of tracking down rogue SQL a bit easier.

11) Other considerations

a) Check DSMAX is sufficiently large - as whilst dataset close yes/no is not as important as it was, forcing DB2 to close datasets unnecessarily should be avoided especially as the maximum value has be lifted from 10,000. Also look at PCLOSET/PCLOSEN ZPARMs that affect the opening and closing of datasets by DB2 as well.
b) Each open VSAM DSCB needs 0.3 MB of below line storage. If the maximum is reached (32,767) 10 MB of below line storage is required, not a good place to be.
c) EDM Pool - Check that its large enough, ensure that there are no occurrences of the EDM Pool being too small to load the necessary objects when they are required. Note that under V6 EDM Pool problems are a lot less likely as DB2 can load the bit of the DBD it needs into the EDM Pool before V6 the entire DBD would have to be loaded.
d) Dynamic Caching - On a system that uses mostly dynamic SQL it may be worth setting the cache on to avoid some prepares. Ensure you understand dynamic caching it fully before you implement it - on a system that is mostly QMF it will almost definitely be a waste of storage. The Dynamic SQL Cache is stored in the EDM pool so if dynamic caching is switched on in an existing DB2 subsystem the EDM pool needs to be increased in size. As well as monitoring this at application level it should also be monitored at system level using the DB2 PM statistics report.
e) Checkpoint frequency - aim for about 15 minutes which can be effected by ‘ altering’ the LOGLOAD ZPARM - this can now me done dynamically in V6.

This is considered because it affects write to DASD for updated pages in bufferpools.

f) .

g) DB2 Connections - ensure adequate values for CTHREAD, MAXDBAT and IDBACK. The latest DB2 versions utilities e.g. LOAD can spawn many sub tasks to attempt to cut elapsed time but can be constrained by lack of threads

h) Consider placement of DSNDB07 workfiles - try and spread across volumes and 'controllers' even with latest disk technology. Check for size of datasets - fix all but the last with zero secondary allocations so that growth can be monitored by number of extents taken by the last workfile. **Note:** in a sysplex environment the work file data base will not be called DSNDB07 but will be given a customisable name. The table spaces in that database will always have a name of the format DSN4Knn.

i) Hiperpools may be of use on DSNDB07 if you have spare storage.

j) Ensure that delete/define of underlying DSNDB07 datasets is performed to reclaim space taken on an occasional basis - it will not be reclaimed any other way.

k) IRLM address space - check that on small systems PC=NO as there will be some performance benefit. For systems with lots of locking ensure PC=YES as this will prevent the storage for locking filling up and the problems this will cause.

l) Lock escalation - Ensure there is some this as this will save storage in IRLM, this will need to be balanced with the need for concurrent update to the same object.

12) **DB2 Objects**

a) The key here is to ensure that the 3 'R's are performed: Resize, Reorg and Runstats - additionally consider rebinds for static SQL.

b) Many performance problems can be caused by excessive I/O caused by badly sized objects, objects in many extents and fragments across many volumes. Poor disorganisation leads to low cluster ratios which lead to indexes not being favoured.

c) These are particularly important for dynamic queries - 'bound' plans/packages will not be affected by degraded objects. Most often seen when people do regular runstats to check the state of organisation / growth but then do not resize or reorganise thus exacerbating the problem.

d) Rebinds after version upgrades/ service packs etc. should be considered - the optimiser does improve and if there is a worry a 'test; bind and explain into a 'test' collection can be tried without disrupting the production 'access path' - and of course we are all using packages and not plans with DBRMs anymore!

e) Consider objects placement as well. Keep DB2 system objects in their own pool - on their own, keep DB2 application data in their own pools, consider spreading tablespaces and indexspaces through the use of the 'SMS' ZPARMS in V6. And of course keep all non-DB2 data out of the pools used by DB2.

f) On DASD, consider placement - ensure work is spread across logical volumes and that those volumes are spread across logical control units. Even with ESS shark and PAV etc. poor 'placement' can place extra load where it need not happen.

TCPIP and FTP

Useful TCP/IP and FTP Commands
http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/tips0091.html?Open

(End of document)