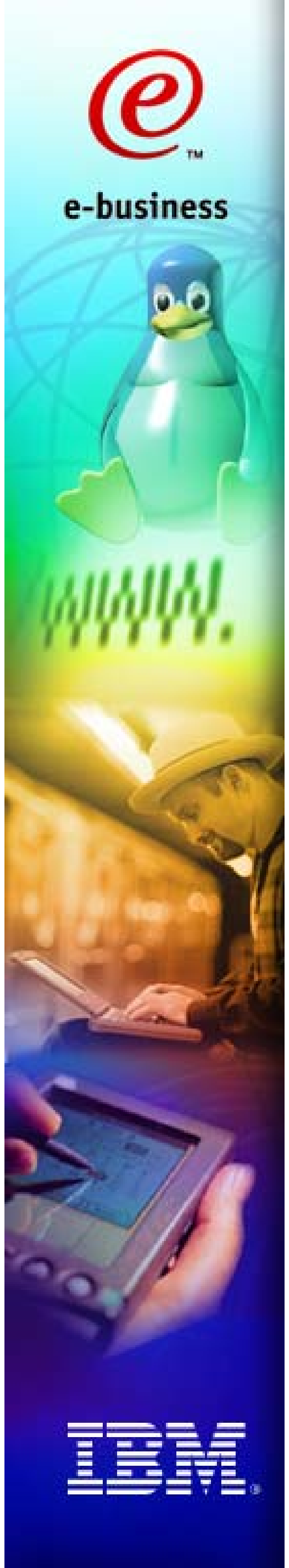


DB2 and Linux on zSeries

A performance problem and its solution

Malcolm Beattie, IBM UK
Linux Technical Consultant, EMEA ESG

Session L801, zSeries Technical Conference, Munich

4 June 2003

Introduction

- DB2, Linux, z/VM, zSeries
  - a performance problem and its solution
  - a real case study
  - simplified a little for this exposition
- Problem statement and initial analysis
- Detour on FCON and detailed analysis
- ESS architecture and configuration
- Solution and results
- Conclusions
- Questions

Problem Statement I

- Customer has DB2 and Linux on an S/390
- Application completed functionality testing and entering system test
- Application does not scale
- Performance at required load is very bad

Hardware and Software

- Hardware
  - 9672-Xn7 (G6)
  - Enterprise Storage Server (Shark) F20
- Software
  - z/VM 4.3
  - Linux 2.4 (SLES7)
  - DB2 V7
  - WAS

Application software

- Runs "streams" of SQL queries
- Each stream is a never-ending sequence of SQL queries
- Each SQL query is (for the purposes of this talk)
  - chosen at random from a huge collection of possible queries
  - read-only (SELECT) almost always
  - programmatically generated
  - complex
  - usually throughput bound rather than latency bound

Problem Statement II

- Performance requirement:
  "keep up with real time"
  - Calculation implies we need four concurrent streams
- Running one stream performs acceptably
- Running another concurrently slows the first
  - Overall throughput is only a bit higher
- Running more than two slugs the rest
  - Overall throughput barely increases at all
- Linux ps shows DB2 processes in D state
  - "D" means process waits uninterruptibly in kernel

Hardware & Software Healthcheck

- Gather information from
  - S/390 administrators
  - VM administrators
  - Linux administrator
  - Database administrator
- Installed FCON performance monitor for VM
  - excellent software for performance analysis
  - will be optional priced feature of z/VM as of V4.4
    - supersedes RTM and PRF (superset of their functionality)
    - will go by the name "VM Performance Toolkit"
  - Third party products available too (e.g. ESAMON)

Healthcheck results

- No obvious major problems with hardware
  - ► Our LPAR has assigned to it
    - – enough memory and CPU (confirmed by FCON)
    - – 12 ESCON channels
    - – "half a Shark" of DASD (~1.5 TB), devices defined as 3390-3
    - – OSA network port
- No obvious major problems with software
  - ► Database size is 100s of GB
  - ► Linux, DB2 and application all functionally OK
- No obvious major configuration problems
  - ► DB2 configuration could be optimised somewhat
  - ► DB2 tables spread over lots of 3390-3 devices
    - – allocated as "almost full-pack" minidisks (cylinders 1-END)
    - – VM MDC (MiniDisk Cache) settings were acceptable

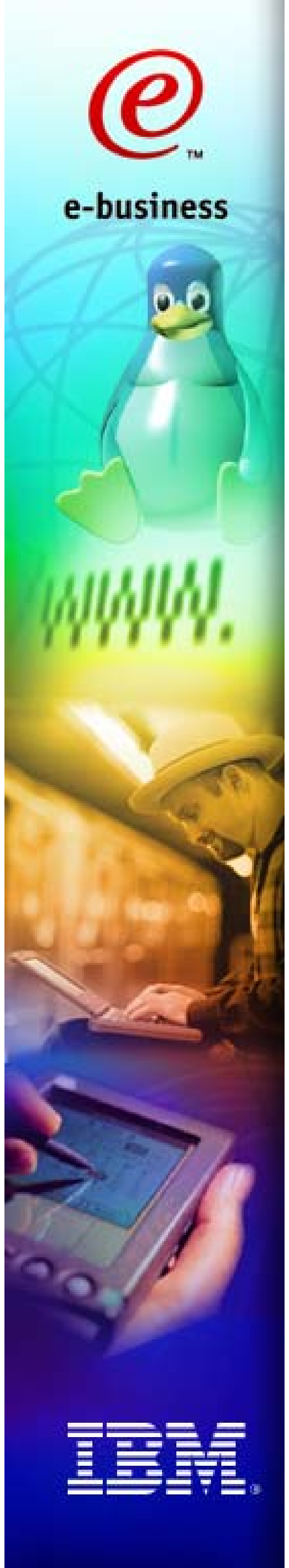
A closer look at FCON

- Find representative "bad" queries
  - tablescans of four of the largest tables
- Memory figures OK: VM is not paging
- CPU figures OK: guest is not saturating CPU
- I/O figures look suspicious:
  - Channel utilisation very high for some channels
    - sometimes 70-80%, anything over 50% deserves a closer look
  - High I/O rates hence I/O subsystem being exercised
  - DASD I/O screen shows awfully high response times
    - some around 60ms, anything over 10ms deserves a closer look
- Why are the DASD response times so high?

Detour on FCON

FCON main menu

```
FCX124          Performance Screen Selection  (V.3.2.05/18)     Perf. Monitor

General System Data          I/O Data                      History Data (by Time)
1. CPU load and trans.       11. Channel load              31. Graphics selection
2. Storage utilization       12. Control units             32. History data files*
3. Storage subpools          13. I/O device load*          33. Benchmark displays*
4. Priv. operations          14. CP owned disks*           34. Correlation coeff.
5. System counters           15. Cache extend. func.*      35. System summary*
6. CP IUCV services          16. DASD I/O assist           36. Auxiliary storage
7. SPOOL file display*       17. DASD seek distance*       37. CP communications*
8. LPAR data                 18. I/O prior. queueing*      38. DASD load
9. Shared segments           19. I/O configuration         39. Minidisk cache*
A. Shared data spaces        1A. I/O config. changes       3A. Paging activity
B. Virt. disks in stor.                                    3B. Proc. load & config*
C. Transact. statistics      User Data                     3C. Logical part. load
                             21. User resource usage*      3D. Response time (all)*
D. Monitor data              22. User paging load*         3E. RSK data menu*
E. Monitor settings          23. User wait states*         3F. Scheduler queues
F. System settings           24. User response time*       3G. Scheduler data
G. System configuration      25. Resources/transact.*      3H. SFS/BFS logs menu*
                             26. User communication*       3I. System log
Select performance screen with cursor and hit ENTER
Command ===>
F1=Help  F4=Top  F5=Bot  F7=Bkwd  F8=Fwd  F12=Return
```



More from FCON...

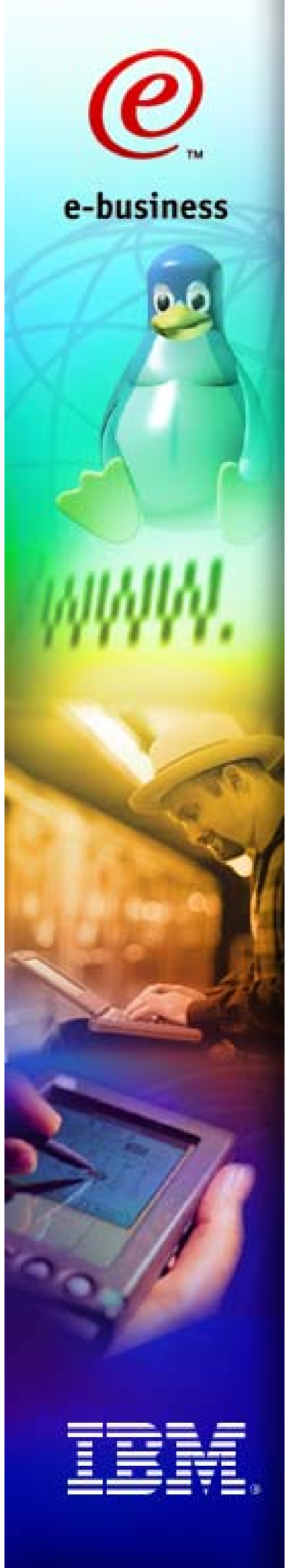
Example "Channel Load" screen

```
FCX107      CPU 2064  SER 111CA  Interval 13:29:03 - 14:07:03    Perf. Monitor

CHPID Chan-Group    <%Busy>     <----- Channel %Busy Distribution 13:29:03-14:0
(Hex) Descr  Qual   Cur  Ave    0-10 11-20 21-30 31-40 41-50 51-60 61-70 71-80
 1A   ESCON   --      6    0    100     0     0     0     0     0     0     0
 33   ESCON   --      3    0    100     0     0     0     0     0     0     0
 AB   ESCON   --      3    0    100     0     0     0     0     0     0     0
 EA   ESCON   --      3    1    100     0     0     0     0     0     0     0
 F7   ESCON   --      3    0    100     0     0     0     0     0     0     0
 02   ESCON   --      0    0    100     0     0     0     0     0     0     0
 03   ESCON   --      0    0    100     0     0     0     0     0     0     0
 07   ESCON   --      0    0    100     0     0     0     0     0     0     0
 08   ESCON   --      0    0    100     0     0     0     0     0     0     0
 11   ESCON   --      0    0    100     0     0     0     0     0     0     0
 12   ESCON   --      0    0    100     0     0     0     0     0     0     0
 14   ESCON   --      0    0    100     0     0     0     0     0     0     0
 1B   ESCON   --      0    0    100     0     0     0     0     0     0     0
 1C   ESCON   --      0    0    100     0     0     0     0     0     0     0
 1D   ESCON   --      0    0    100     0     0     0     0     0     0     0
 20   ESCON   --      0    0    100     0     0     0     0     0     0     0
 23   ESCON   --      0    0    100     0     0     0     0     0     0     0

Command ===>
F1=Help  F4=Top  F5=Bot  F7=Bkwd  F8=Fwd  F10=Left  F11=Right  F12=Return
```

More from FCON...

Example "I/O Device Load" screen

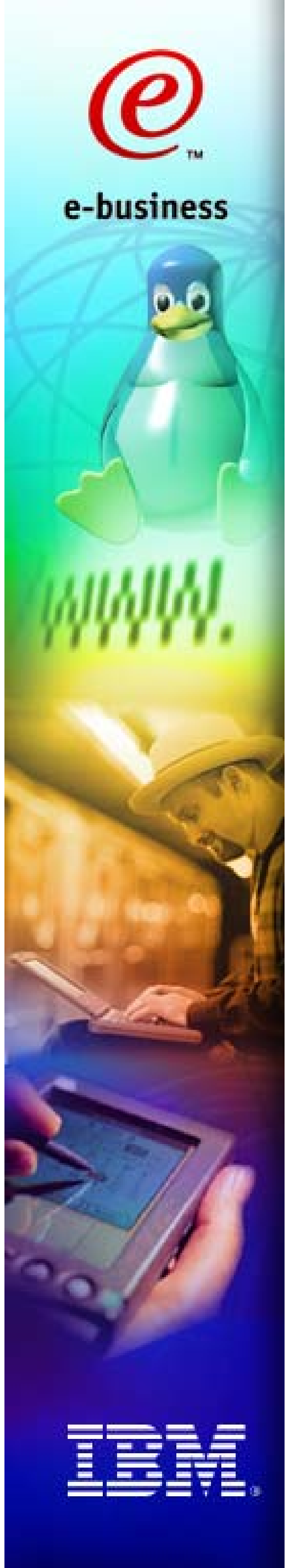
```
FCX108      CPU 2064   SER 111CA   Interval 14:24:03 - 14:25:03    Perf. Monitor
.    .                    .         .     .    .    .    .    .  ____    .    .
<-- Device Descr. -->  Mdisk Pa-  <-Rate/s-> <------- Time (msec) -------> Req.
Addr Type   Label/ID   Links ths   I/O Avoid Pend Disc Conn Serv Resp CUWt Qued
>> All DASD <<          ....        .0    .0   .1   .0   .4   .5   .5   .0  .00
020A CTCA   >LINUX8      ...    1   .3   ...   .1 3000  1.0 3001 3001   .0  .00
0202 CTCA   >TCPIP       ...    1  1.7   ...   .2  600   .1  601  601   .0  .00
1000 3390-3 VMLX5A CP     57    8   .1    .0   .1   .0  1.4  1.5  1.5   .0  .00
0190 3390-3 MNT190         0    4   .0    .0   .2   .0   .4   .6   .6   .0  .00
0191 3390-3 CMS191         0    4   .0    .0   .2   .1   .3   .6   .6   .0  .00
0592 3390-3 TCM592         0    4   .0    .0   .2   .0   .4   .6   .6   .0  .00
2303 3390-3                0    8   .0    .0   .3   .0   .3   .6   .6   .0  .00
019D 3390-3 MNT19D         0    4   .0    .0   .1   .0   .4   .5   .5   .0  .00
019E 3390-3 MNT19E         0    4   .0    .0   .1   .0   .4   .5   .5   .0  .00
1002 3390-3 VMLX5C         5    8   .0    .0   .2   .0   .3   .5   .5   .0  .00
2101 3390-3 0X2801         0    8   .0    .0   .1   .1   .3   .5   .5   .0  .00
2102 3390-3 0X2804         0    8   .0    .0   .1   .1   .3   .5   .5   .0  .00
2103 3390-3 0X2204         0    8   .0    .0   .1   .1   .3   .5   .5   .0  .00
2104 3390-3 0X5000         0    8   .0    .0   .1   .1   .3   .5   .5   .0  .00
2200 3390-3 LNX10X         0    8   .0    .0   .1   .1   .3   .5   .5   .0  .00
2201 3390-3 LNX10X         0    8   .0    .0   .1   .1   .3   .5   .5   .0  .00
Select a device for I/O device details
Command ===>
F1=Help  F4=Top  F5=Bot  F7=Bkwd  F8=Fwd  F10=Left  F11=Right  F12=Return
```



More from FCON...

Example "Control Units" screen

```
FCX176      CPU 2064  SER 111CA  Interval 14:25:03 - 14:26:03    Perf. Monitor
____ .            .     .     .     .     .     .     .     .     .     .     .
Sub-          <-----Cache Size------> <----------- DASD Load Data ------------>
sys-          <Volatile-> <Non-Volat> <--Total-->
tem  Control <--(MB)---> <--(kB)---> <I/O rates>  Pct <----- Time  (ms) ------>
ID   Unit     Conf Avail NV-Cf NV-Av Cache SCMBK Busy Pend Disc Conn Serv Resp
1152 9393 RV  1024  1024  8192  8192    .0    .0    0   .1   .0   .4   .5   .5
1153 9393 RV  1024  1024  8192  8192   1.8    .1    0   .1   .0   .4   .5   .5
2482 9393 RV  2048  1497  8192  8192    .1    .8    0   .1   .1   .4   .6   .6








Command ===>
F1=Help  F4=Top  F5=Bot  F7=Bkwd  F8=Fwd  F10=Left  F11=Right  F12=Return
```



More from FCON...

Example "Cache extend. func." screen

```
FCX177      CPU 2064   SER 111CA   Interval 14:27:03 - 14:28:03    Perf. Monitor
____ .      .       .      . . . .        .      .      .      .      .    .    .    .
<--Device Descr.-> Stg   C D D     <--------- Rate/s ----------> <--------------
                   Ctlr A F U      Total Total  Read  Read Write      <------ Hi
Addr Type   VOLSER ID   C W L ST   Cache SCMBK N-Seq   Seq    FW Read Tot RdHt W
0190 3390-3 MNT190 1153 A A - 00      .1    .0    .1    .0    .0  100 100  100
0191 3390-3 CMS191 1153 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
019D 3390-3 MNT19D 1153 A A - 00      .1    .0    .1    .0    .0  100 100  100
019E 3390-3 MNT19E 1153 A A - 00      .1    .0    .1    .0    .0  100 100  100
0592 3390-3 TCM592 1152 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
1000 3390-3 VMLX5A 2482 A A - 00      .1    .1    .0    .0    .1    0 100   .. 1
1001 3390-3 VMLX5B 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
1002 3390-3 VMLX5C 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
1003 3390-3 VMLX5D 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
1004 3390-3 VMLX5E 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
2100 3390-3 0X2800 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
2101 3390-3 0X2801 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
2102 3390-3 0X2804 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
2103 3390-3 0X2204 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
2104 3390-3 0X5000 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
2200 3390-3 LNX10X 2482 A A - 00      .0    .0    .0    .0    .0   ..  ..   ..
See also CACHDBSE for a cache performance summary
Command ===>
F1=Help  F4=Top  F5=Bot  F7=Bkwd  F8=Fwd  F10=Left  F11=Right  F12=Return
```

Analysis

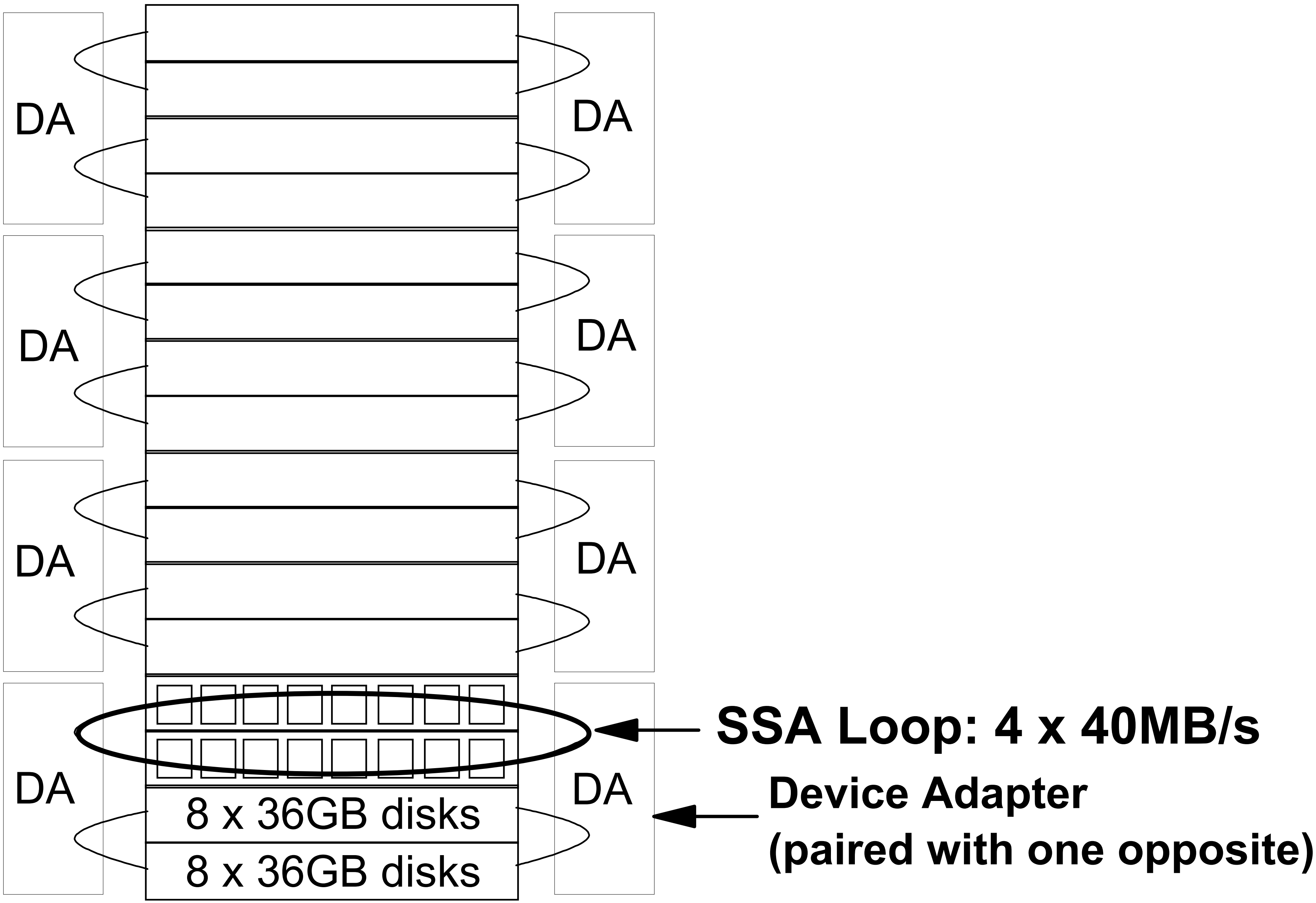
- Bottleneck is at ESS control unit (LCU) level
- Ran two long SQL queries (Q2 & Q3)
- Only 4 of the 8 Shark LCUs were active at all
- During Q2, 1 CU busy, others idling
- During Q3, 1 CU busy, 2 light, others idling
- Running two Qs saturates 1 LCU

Deduce: bottleneck is the way data is spread across LCUs

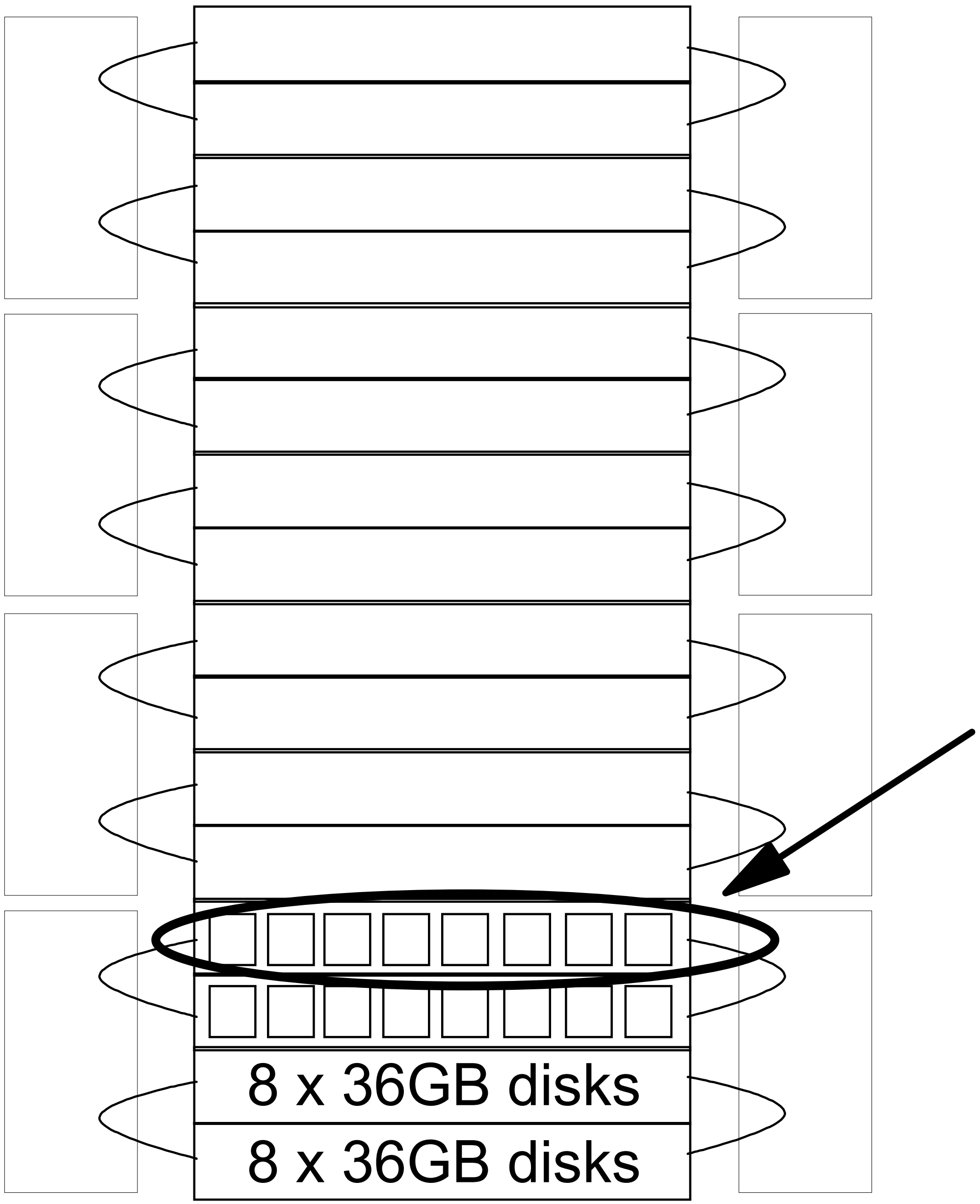
Why do LCUs affect performance?

- ESS shields you from most performance issues
  - Data is spread using RAID
    - good reliability
    - good random I/O
  - Data is cached in memory
    - Large read cache memory
    - Large non-volatile write memory
  - Staging/destaging algorithms optimise reads/writes
  - High-performance back-end adapters and disks
- But you can't *always* treat it as a big black box
- Sometimes you need to know a little more...

A logical view of this Shark

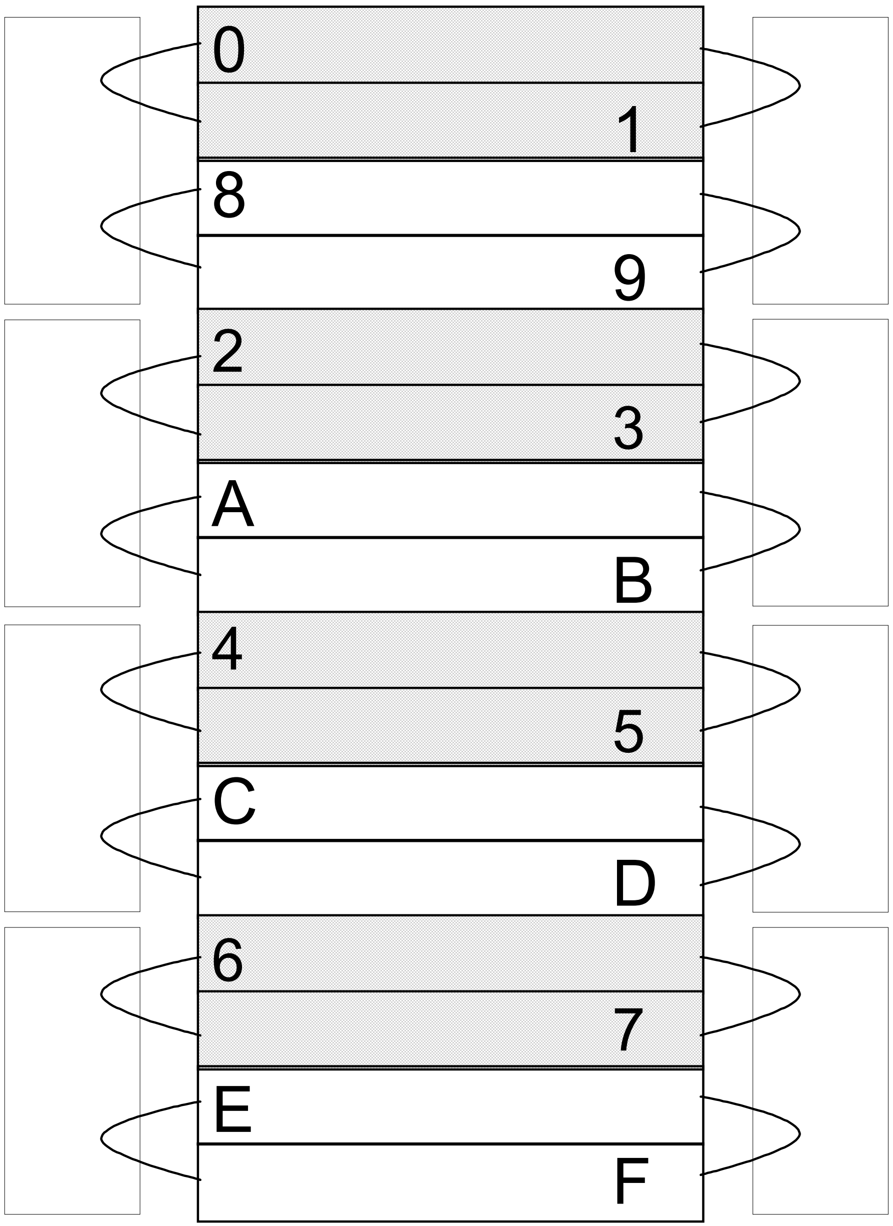


A logical view of this Shark

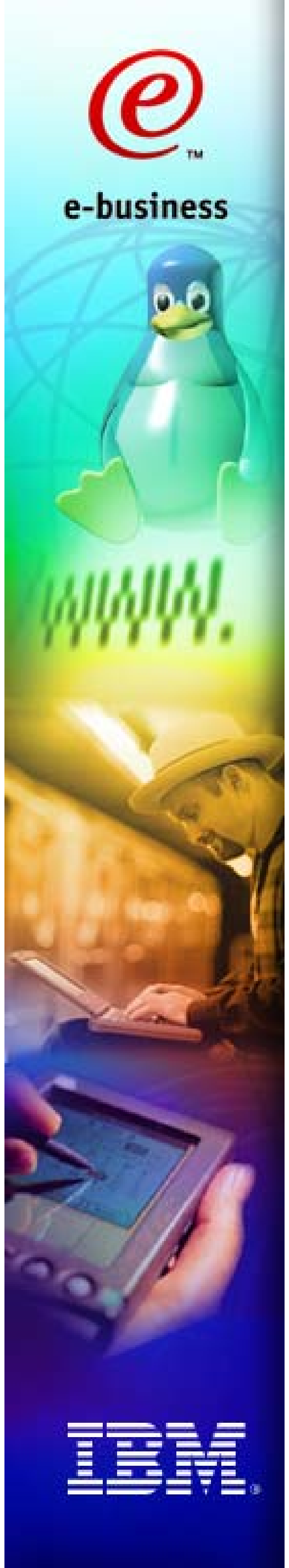


1 LCU
= 8 physical disks
+ 40 MB/s read
+ 40 MB/s write

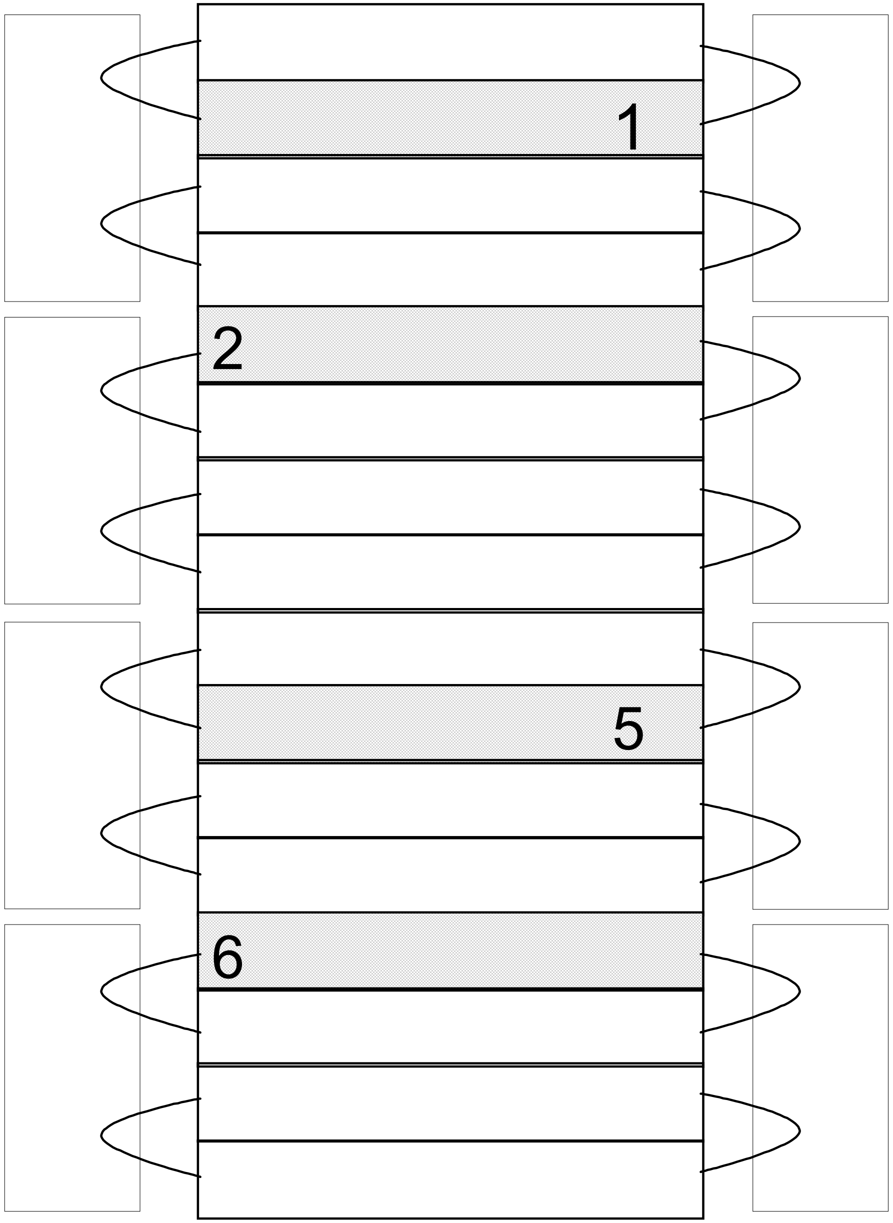
LCU and device numbering



- 1 LCU = 1 LSS (Logical Subsystem)
- LCU number fixed by physical row
- CUADDRs shown
- Good to assign device numbers encoding LCU
- In our case
  - 70XX are on LCU 0
  - 71XX are on LCU 1
  - 72XX are on LCU 2
  - etc
- Our LPAR has 70XX through 77XX



Where does our data live?



- No data was outside LCUs 1, 2, 5, 6
- DB2 tablespaces were spread very unevenly
- Most of the data was in LCU 6
- Most of the rest was in LCU 2

So what was happening?

- Starting one query kept LCU 6 very busy
  - It had 40MB/s read bandwidth and used most of it
  - Performance was good
- Starting a concurrent query caused contention
  - One of the other queries lived mostly on LCU 2
  - It ran fairly well
  - The others lived mostly on LCU 6
  - They ran very poorly
- Starting another concurrent query was dire
  - LCU 6 was saturated

But why was the data so unevenly spread?

Detour on device names and numbers

- ESS
  - Each 3390-3 has a real device number 7nXX
  - From this we can tell which physical row it lives on
  - ESS RAID spreads data horizontally across the 8-pack
  - But it does not spread vertically across LCUs
- z/VM
  - Each real device number 7nXX has a volume label
  - Usually volume labels are chosen to encode the rdev
  - Example scheme: assign label AB7nXX to rdev 7nXX
  - But in this case they were "random"
  - They did not encode the rdev

Detour on device names and numbers

- **z/VM guest directory entry**
  - Each guest virtual machine has virtual devices
  - The *user directory* allocates real resources to guests
  - The user directory contains lines for our Linux guest
  - Each line listing a DASD device for our guest includes
    - a chosen virtual device number
    - the volume label of the real device to be used
    - the cylinder extent (here: cyls 1-END for "nearly full pack" volumes)
  - In our case, the virtual device numbers were "random"
- **Linux**
  - Allocates a numbered list of "slots" for DASD devices
  - The list can be appended to dynamically
  - Each slot is allocated the next device name of the form
    - /dev/dasda ... /dev/dasdz
    - /dev/dasdaa ... /dev/dasdzz
    - /dev/dasdaaa ... /dev/dasdzzz

Detour on device names and numbers

- **Linux filesystems**
  - **In our case, volumes for DB2 are mounted as filesystems**
  - **/dev/dasdab1 is mounted on /db2/dasdab1**
  - **(actually a longer pathname but same idea)**
- **DB2**
  - **Puts its data in tablespaces**
  - **In this case, we use files to back tablespaces**
  - **Tablespaces are created with**
    - **a chosen name**
    - **a list of filenames across which to spread the data**
  - **In our case, tablespace foo is created in files such as**
    - **/db2/dasdab1/xyz-foo-xyz, /db2/dasdac1/xyz-foo-xyz, ...**

How can the DBA find the data?

- The DBA knows the tablespace name
- The DDL creating it shows the filenames
- From the filename,
  the prefix shows the mountpoint
- From the mountpoint,
  /etc/fstab shows device name /dev/dasdab
  - In our case, it is already encoded in the mount point
- From the device name,
  /proc/dasd/devices shows the device number
  - This is the virtual device number for this guest
- Now what to do?

From virtual device to real device

- Now we have the virtual device number
- To go further, we need z/VM CP access
  - either by the Linux administrator
    - logging on to the guest console
    - or using the cpint kernel module to allow CP access from Linux
    - CP QUERY VIRTUAL DASD shows vdev, rdev and volume label
  - or by the z/VM administrator
    - using privileged CP commands
    - or looking up the guest's user directory entry and using QUERY DASD
  - The real device number determines its physical location
- But what could we do without z/VM CP access?

Suggested device names and numbers

- We could encode the rdev in the label
  - z/VM admin can use the user directory entry directly
  - A CP QUERY DASD is no longer needed
  - Tools such as FCON can display the information too
  - Linux admin can still use CP QUERY VIRTUAL DASD
- We could encode the rdev in the vdev
  - DBA no longer needs Linux or z/VM admin to find the rdev
  - But some allocation strategies preclude this
  - An external mapping could be used instead
    - Keep track of vdev allocations
    - Use fdasd labelling, filesystem labelling or a piece of paper...
    - Working with ranges simplifies management
    - Fragmentation by any other name...
- Namespace management always matters

Channel utilisation

- 12 ESCON channels from our LPAR to ESS
  - 6 channels configured to LCUs on one side of ESS
  - 6 channels configured to LCUs on other side of ESS
- Would not comfortably sustain four concurrent heavy tablescans
  - Pend times measured by FCON confirmed this
  - Rough rule of thumb 12MB/s seq read per channel
  - Linux rather greedy for I/O resources
  - Linux channel programs are
    - DEFINE EXTENT
    - LOCATE RECORD
    - from 1 to 32 chained 4KB READ CCWs (or WRITE for writes!)
  - z/VM's CP TRACE IO confirmed 128KB reads
    - Calculate ratio of READ CCWs to SSCH instructions
    - SSCH done by Linux kernel, not apps: no concept of "IOSQ"

Solving the problem

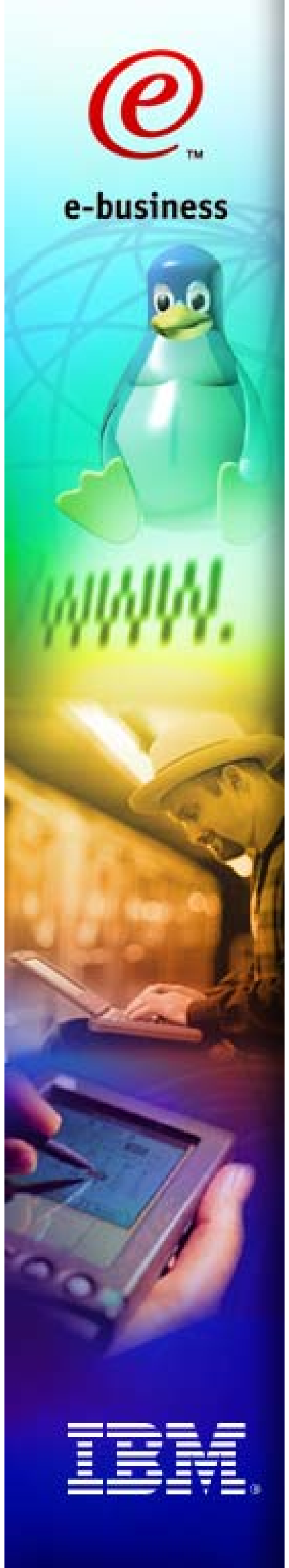
- Number of channels increased to 24
  - 4 sets of 6 paths
  - 2 sets to one side of ESS, 2 to other side of ESS
  - So each real device had 6 paths
  - and heavy reads to 4 LCUs would not cause saturation
- Tablespaces were reorganised
- Main tables spread evenly across 4 LCUs
  - Practical reasons precluded spreading all tables
  - "Manual" spreading: LVM not supported for DB2 V7

Results

- Running one query performed well
  - four lightly used LCUs, not one busy LCU
- Running another concurrent query did too
  - four averagely used LCUs, not one saturated LCU
- Running a third concurrent query did too
  - four LCUs getting busy
- Running a fourth concurrent query did too
  - four LCUs working hard
- Performance now good
  - About four times the total throughput at the start

Conclusions

- Linux DB2 can perform well
- zSeries hardware can perform well
- ESS can perform well
- Sometimes they perform well out of the box
- Sometimes the black box needs to be opened
- Measurement tells you where to look
- zSeries and z/VM provide excellent tools
  - CPU, memory, I/O
  - per-device pend, disc, conn, busy, cache hit, seq, ...
  - FCON (and similar): real-time, history, trends, ...
  - CP TRACE IO for virtual devices
  - CP TRSOURCE ID FOO TYPE IO DEV rdev ...



Questions?

- **IBM top-level page for Linux for zSeries**
  http://www.ibm.com/zseries/linux

- **Linux-390 mailing list (hosted at Marist)**
  http://www.marist.edu/htbin/wlvindex?linux-390

- **Contact Details**

  Malcolm Beattie
  beattiem@uk.ibm.com

- **Thank you!**