



VAX to Integrity in One Jump

Jeffrey S. Jalbert
JCC Consulting, Inc.



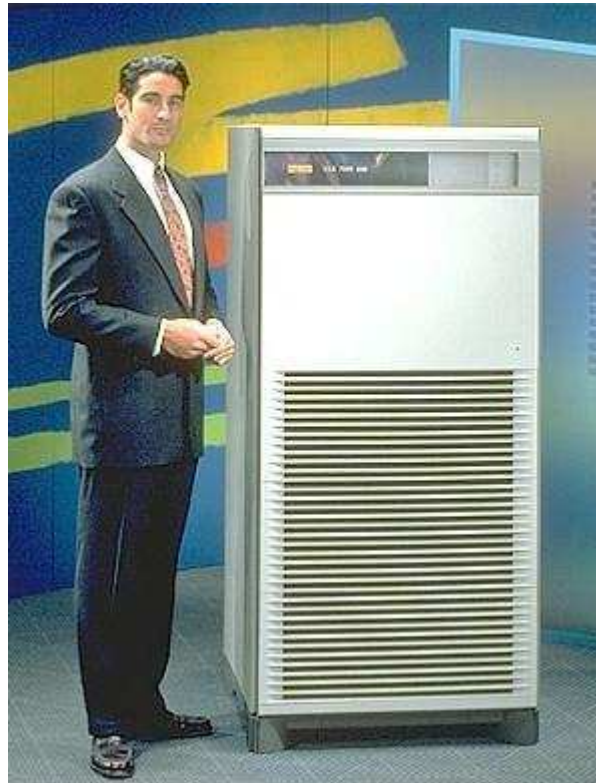
Agenda

This presentation discusses a migration from a VAX/VMS directly to an Integrity OpenVMS platform. The time jump is 14 years and a good bit of progress has happened in those years.

There was concern about how disruptive the change would be. JCC was asked to smooth the path.



Do You Remember When Your Computer Was Bigger Than You?





And Your Cluster Filled a Room?

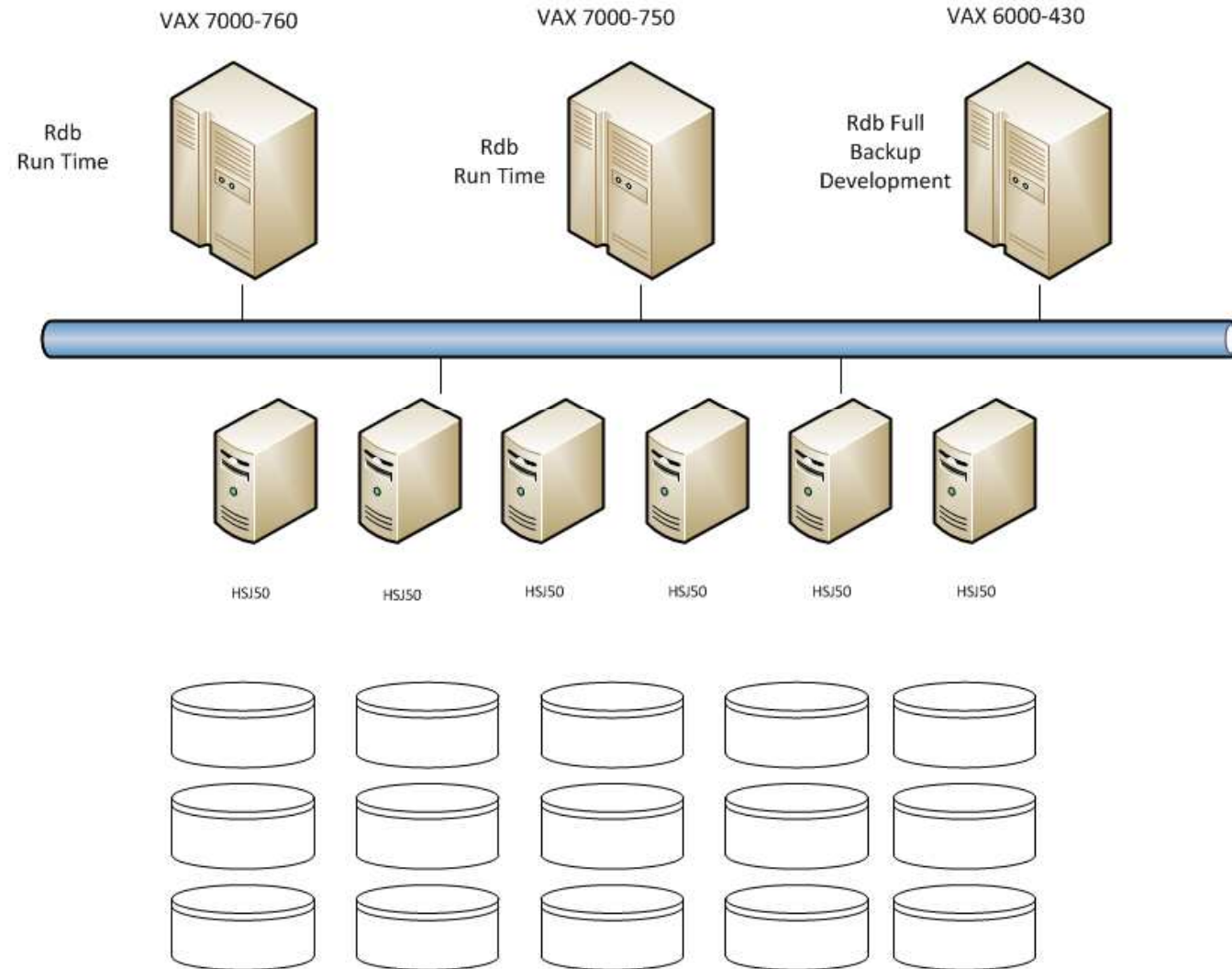


9/2/2011

Copyright 2010, JCC Consulting Inc. All rights reserved.



Production/Development Cluster





Hardware Configurations

- 7000-750 – 768 Mb
- 7000-760 – 3584 Mb
- 6000-430 – 512 Mb



Standby Platform Was VAX 7000





Base Software Platform

- VAX/VMS 7.1
- Rdb 4.1-0
- UCX V4.2
- Others
 - e.g. Hitman
- 296 print queues (world-wide)
- 44 batch queues
- 10 distinct Rdb databases
- Many RMS ISAM and Relative files



Development Environment

- CDD V5.0
- Fortran V5.7
- Cobol V5.3
- TDMS 1.9A
- CMS V3.4
- Rdo



Business Problem

- VAX cluster hardware was old and expensive to operate
- Could not meet the expected workload demands as business expanding
- Software constrained, e.g. no ODBC connections to the database



Elected Solution – rx3600



- Standby machine is an rx2660



Hardware Configuration

- 16 Gb
- 1 dual-core CPU (1.67 GHz/9.0Mb)
- 2 HP smart-array disk controllers
 - Each with 8 15K Rpm drives, 140 Gb each
 - Raid 1 for 1 virtual device
 - Raid 5 for 1 virtual device
 - 1 Spare
- Shadowing across controllers



Software Configuration

- Open VMS 8.3 1h1
- TCPIP V5.6 ECO 4
- Rdb V7.2-400
- Volume Shadowing
- Samba V 1.1-1
- Apache
- Tomcat
- Java, Perl, PHP



Development Environment

- CMS V4.6
- Fortran 8.2-0
- Cobol V2.9-1453
- TDMS V2.0-ECO1
- Rdo and SQL
- SQL Services V7.3-020



System Management

- VMS setup on Integrity was relatively trivial
 - Queues moved by simply editing result of show queue/full on VAX
 - Disk configuration was done with on-board utilities
 - Spare disk was configured with MSA\$UTIL
- Copied the authorization and rightslist files
 - Increased working set quotas and ENQLM
- Defined all logical names relative to the new disks
- Increased MAXPROCESSCNT



Conversion Challenges

- Moving databases
- Moving RMS files
- Building Application



Moving Databases

- Version jump (4.1 to 7.2.4) is too big for RMU/CONVERT
- Export-import required
 - Export on VAX requires hours
 - Import of 1 database on VAX requires a day
- Rdo is legacy tool, does not support current database capabilities
 - All import scripts converted to SQL and JCC default physical database design applied



Converting Databases

- Export on VAX
 - Not fast
- Copy across DECnet network
 - Performance limited by bandwidth of the 3 Ethernet controllers on the VAXes
- Import all 10 databases concurrently
 - Required 2 hours wall time
- Journals placed on system disk



Database Configuration

- All database buffer sizes were standardized to 24 blocks
- Most important database was given 40,000 global buffers after two weeks of production
 - Reduced I/O per transaction from 70 to 5
- Index node sizes standardized to 480 or 960
 - Duplicates are ranked, otherwise traditional sorted



RMS Configuration

- RMS files copied as they existed
- Built a script to “optimize” indexed files
 - Generate FDL
 - Optimize FDL for flatter files
 - Edit to add fill (90%)
 - Reduced file sizes by 75% (they had never been reorganized)
- Not applied yet because performance is so good



Moving the CDD

- All TDMS and record definitions are in the CDD.DIC portion of the dictionary
- Copy CDD.DIC via DECnet to new system



Moving CMS Library

- VAX version of CMS library was old
 - The format of this library must be converted to current internal form
 - Could not be converted because library was corrupt
- Workaround
 - Mark all elements to include history in header/footer when extracted
 - Extract all elements and copy to Integrity
 - Insert all elements in new library



Compiling Code & Building .OBJ Libraries

- New compilers enforce syntax more strictly
 - A couple of edits to Fortran code were required
- Fortran modules are all included in a single source file
 - When inserting into object library get one module
 - Required /SEPARATE_COMPILATION qualifier



Linking Issues

- Multiple modules containing Rdb code were issue when linked into same image
 - All were trying to initialize various database handles
 - Fixed by compiling all but one module with `/noinitialize_handle`
 - Because of the way code modules were used, required introduction of single module to attach to databases



Run Time Data Issues

- LIB\$WAIT
 - Compile 1 COBOL module with the /FLOAT=G_FLOAT qualifier

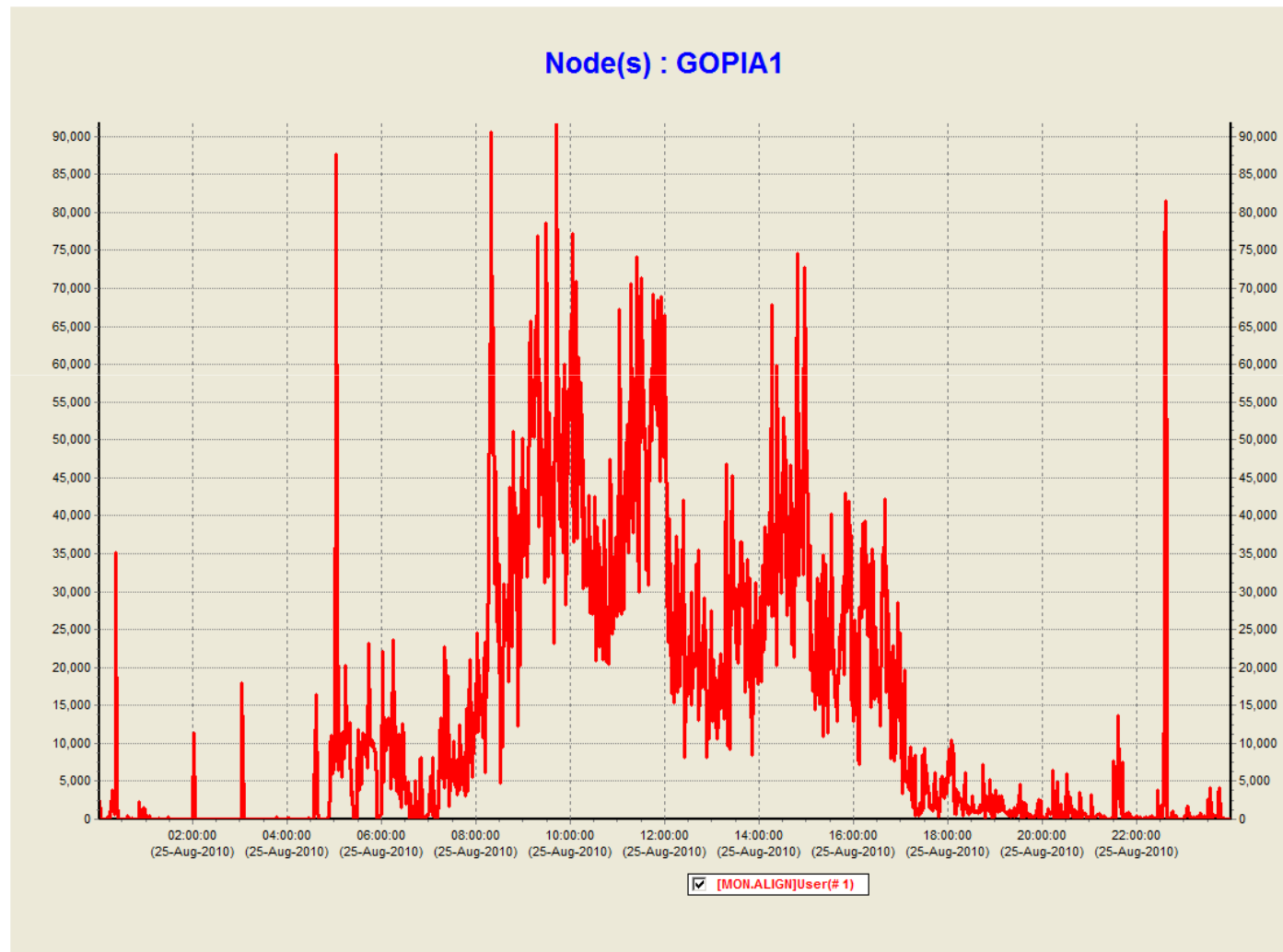


Cutting Over

- Cutover was a bit traumatic
- All data and database export files were copied via the network
- 3-10 Mbit network ports used on the VAX
- Took the better part of a day
- Production was in read-only mode



Run Time Issues – Alignment Faults



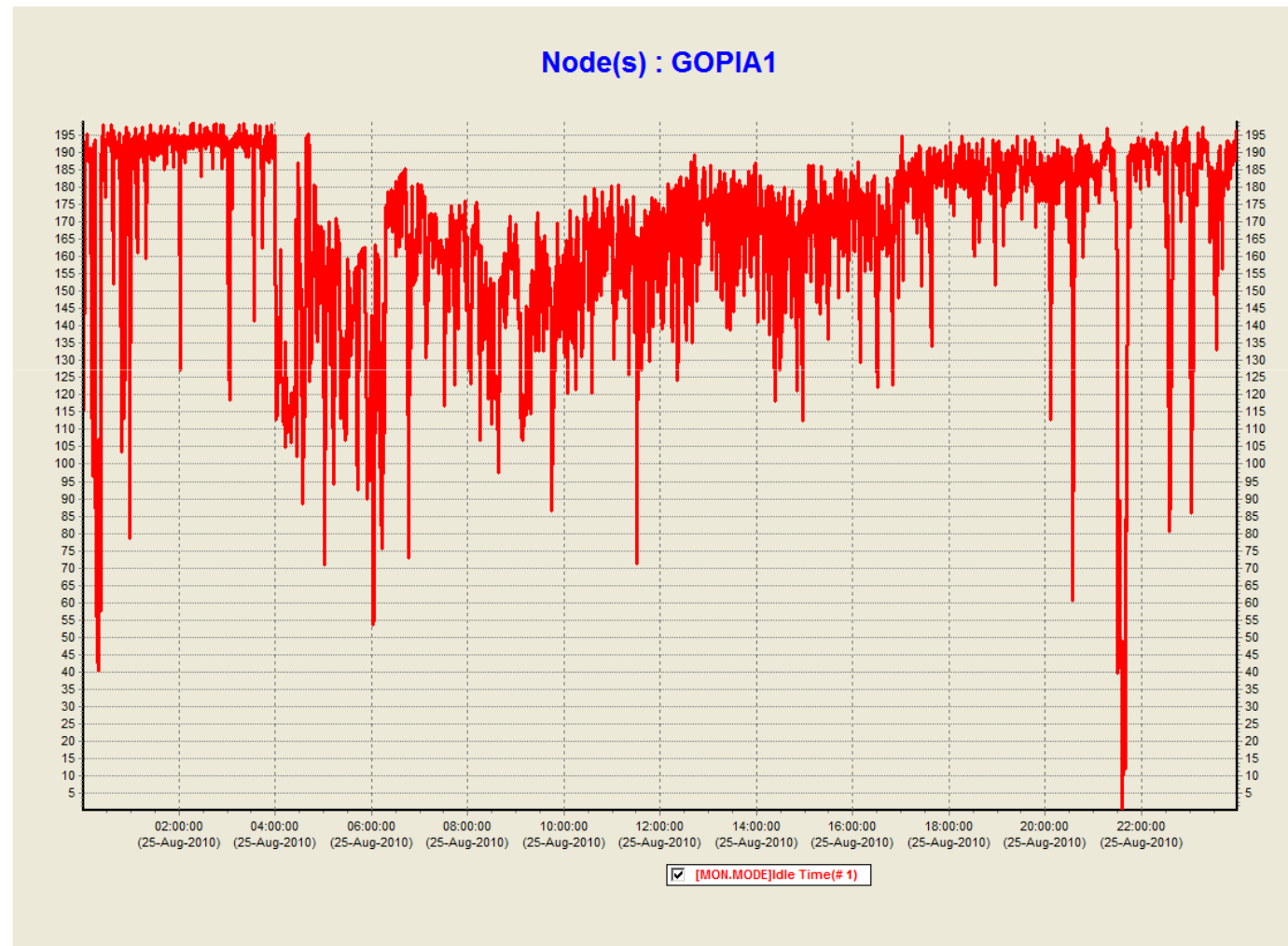


Alignment Faults

- User-mode alignment faults traced to interactive processes
 - Records were badly unaligned
 - Fixing record definitions did *not* fix problems
- Did not affect batch jobs, the compilers did their job of detecting unaligned data
- Using SDA, found that all user-mode alignment faults were in TSSSHARE (TDMS)
 - Issue still being worked with HP support

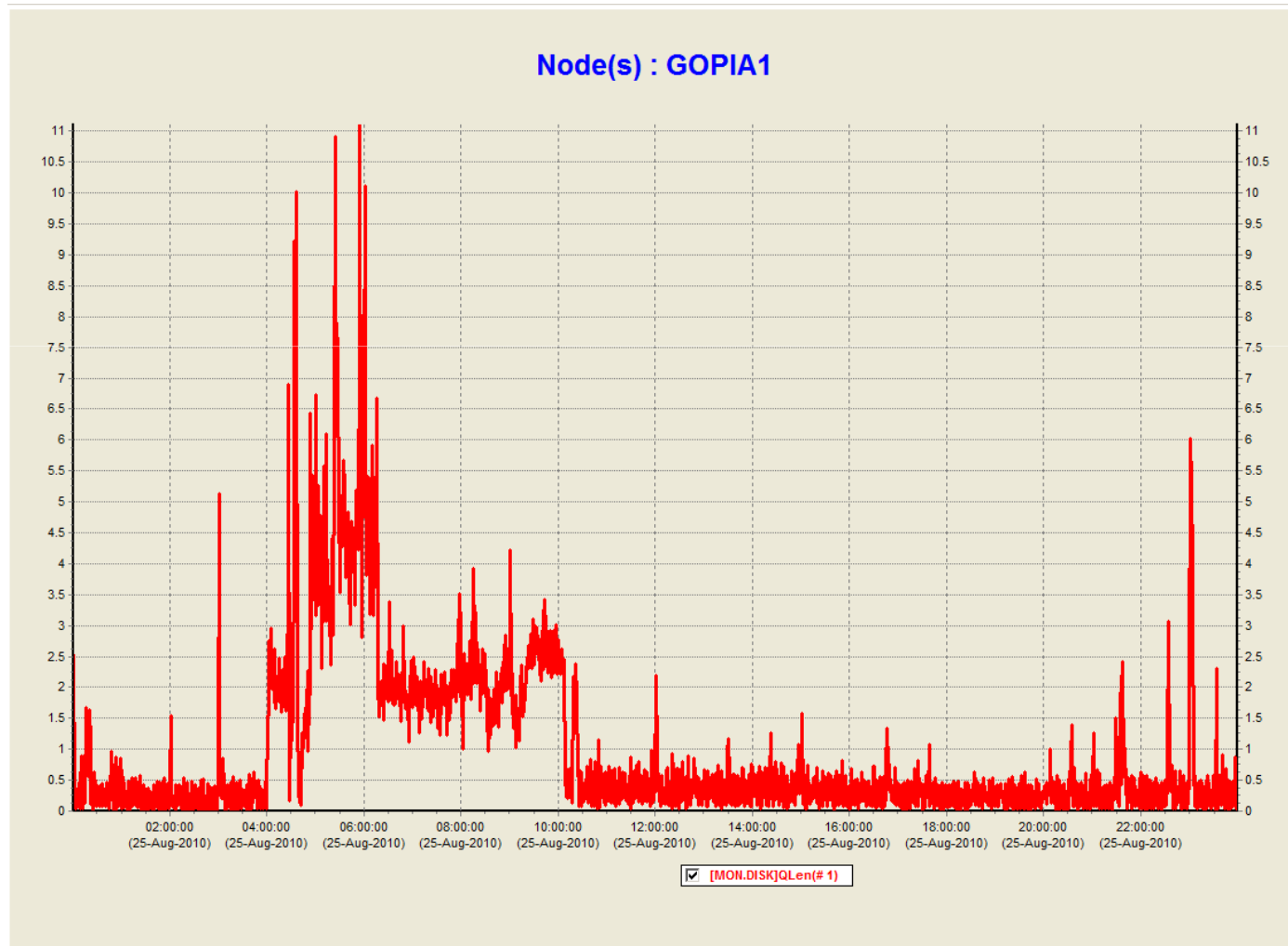


Some Performance Data – Idle CPU





Some Performance Data – Disk Queues



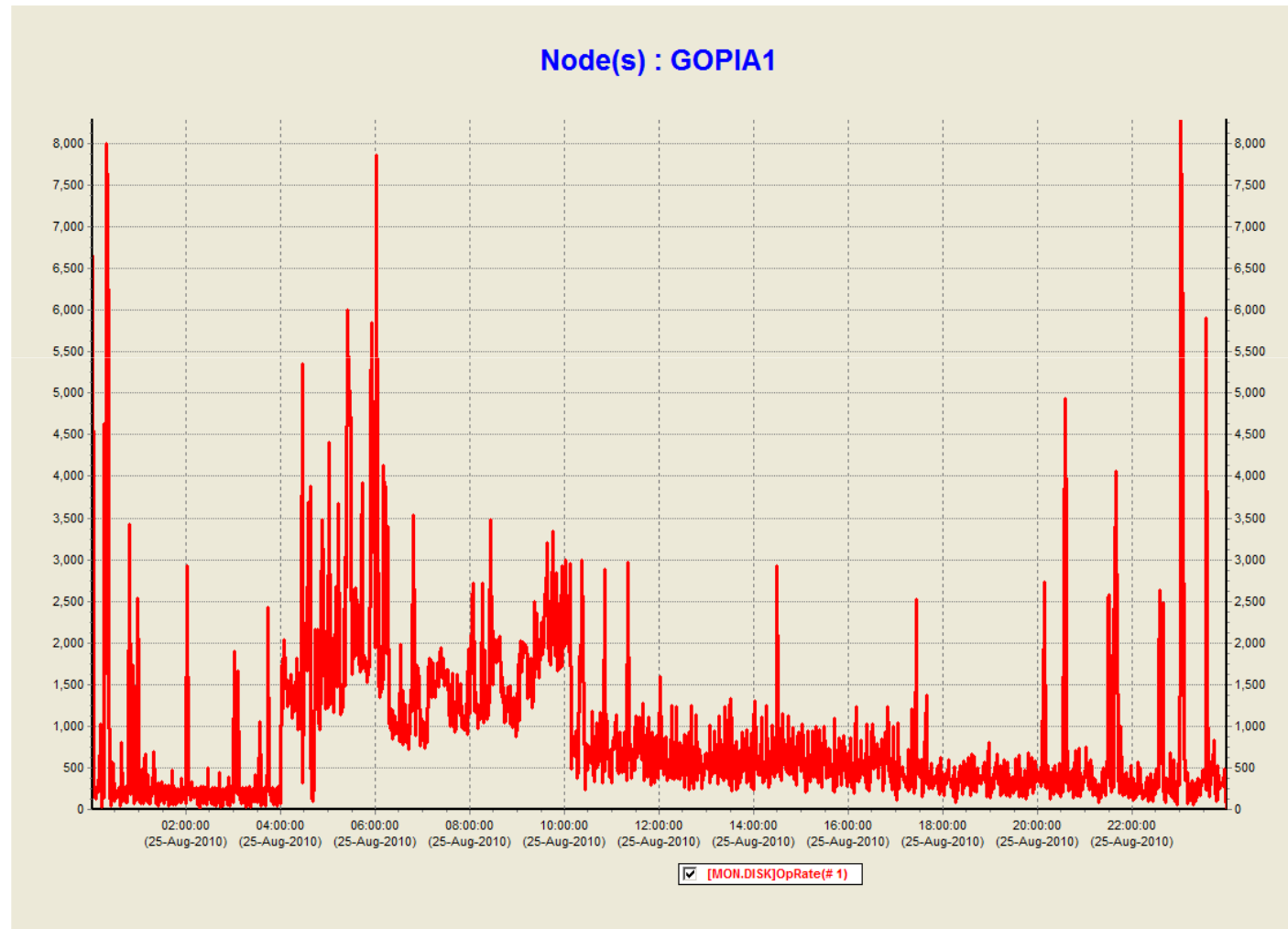


Disk Queues

- Disk queues are due to
 - Backup
 - Nightly batch jobs
- Essentially zero during the day

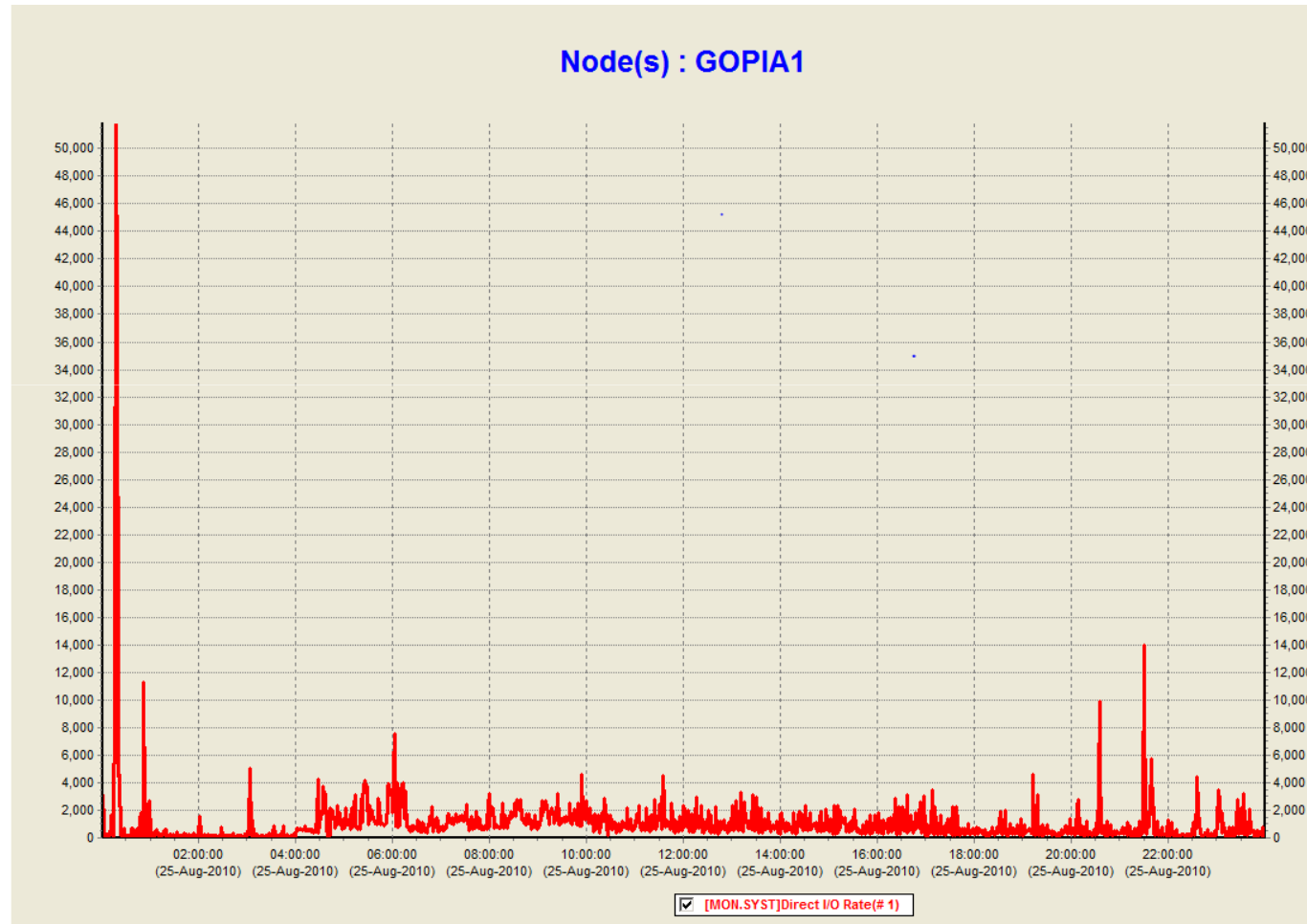


Disk Operation Rate





DISK I/O Rate





Memory Utilization

- XFC cache helps address RMS issues well.
 - 7.91 Gb used for cache
 - 97% read hit rate (over 2 months of up-time)
 - Swallows the I/O demand of the application
- 40,000 global buffers are used for 1 database, otherwise local buffers are used
- Plenty of free memory (3 ½ Gb) when system fully loaded during the day



Performance Comparison

Daily Job	VAX	Integrity
AP_NIGHTRUN	3 to 4 hours	< 9 minutes
AR_DSO_CALC	3 to 4 hours	< 4 minutes
INTL_OVERNIGHT_JOB	6 to 8 minutes	< 1 minute
OVERNIGHT_JOB_1	40 to 50 minutes	< 2 minutes
OVERNIGHT_JOB_2	3 to 4 hours	< 4 minutes

Relative performance gain is between 30x and 60x



Anecdotal Performance Story

- End of Month job:
 - Submits the real work
 - Work jobs run.
 - At end, broadcast message that they are done
 - Waits for a bit
 - Broadcasts message that work has been submitted
- Broadcast messages occurred out of order the first time this process was run!



Conclusions

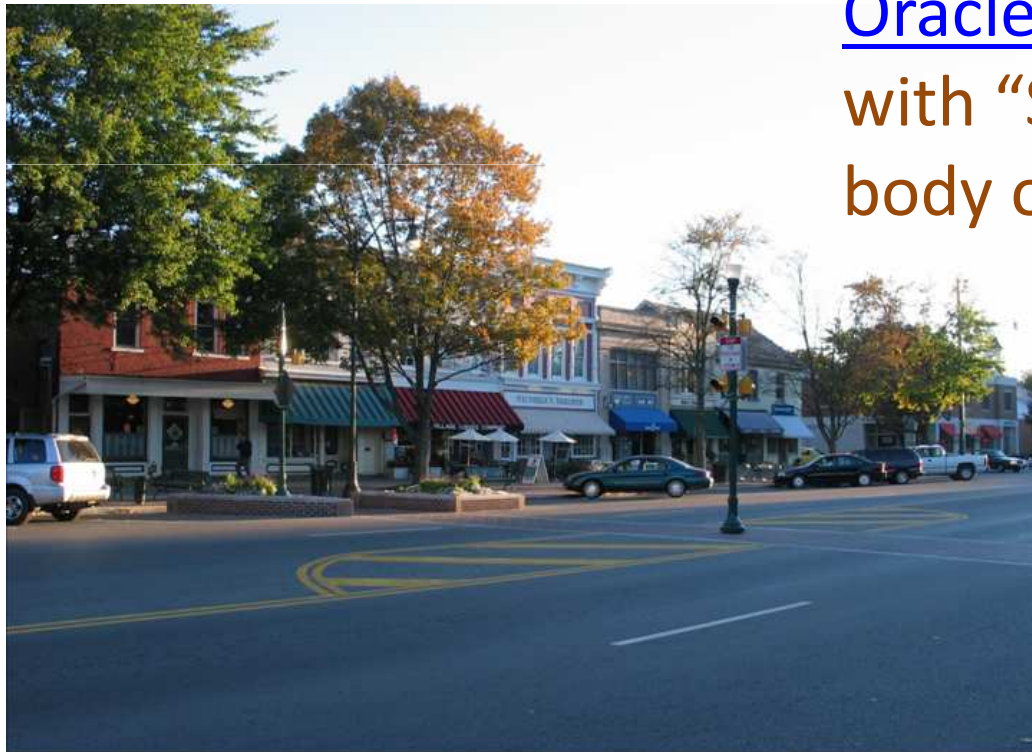
- Code conversion was straightforward, only a couple of edits of Fortran syntax
- Some (small) program redesign because of Linking issues
- CDD was trivial, just copy the file
- Compilation of Rdb modules & the Linker issues required greatest care
- TDMS is the most problematic issue at the moment.



Join the Conversation

Join the worldwide Rdb community. Send mail to

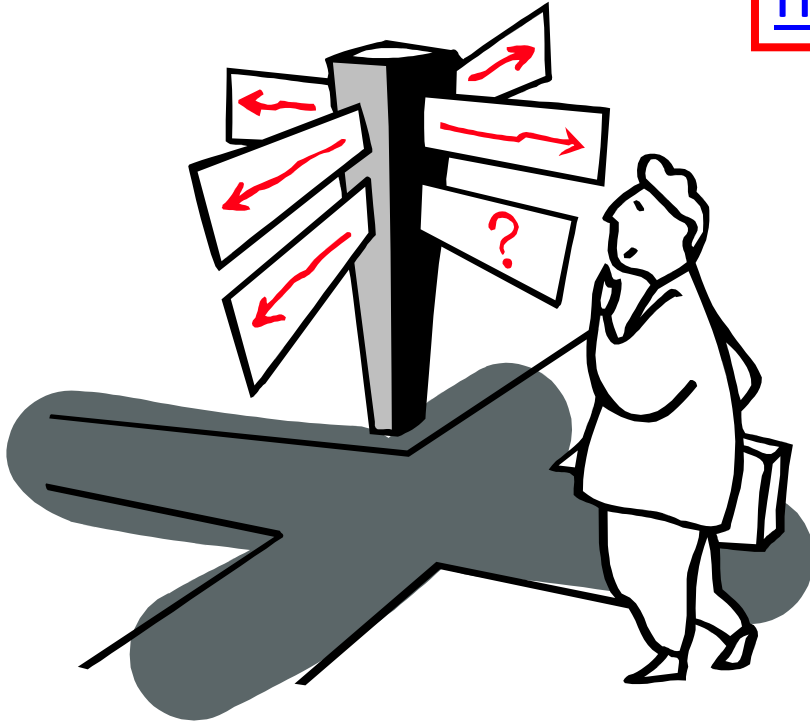
OracleRdb-request@JCC.com
with "SUBSCRIBE" in the
body of the message.





Questions?

<http://www.jcc.com>



Send your input and requests to Info@JCC.com