Superdome performance

Partitions

Dynamic CPU Allocation

Other Techniques

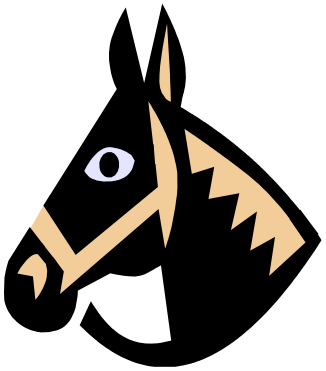
Tom Anderson
HP

Contents

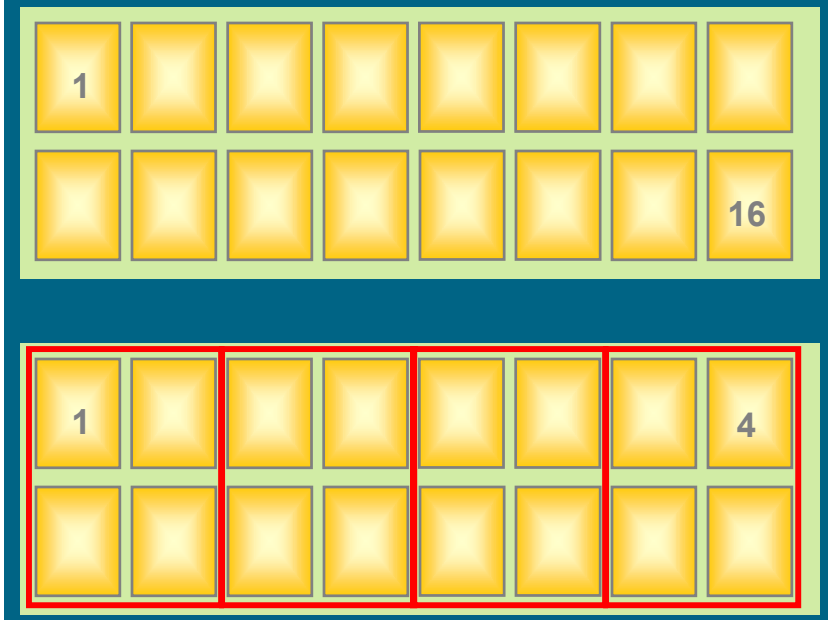
- Hard partitions and virtual partitions
- Tuning advantages of partitions
- Software performance optimizations
- Mixing processors in a Superdome
- Instant capacity on demand (iCOD)
- Dynamic CPU allocation options
- Variable usage (pay for use)
- HP-UX & Windows & Linux all in the same Itanium Superdome (future)
- Summary

Hard partitions and soft partitions

- Hard partitions (nPartitions)
  - "separate" servers but without separate "skins"
  - Electrical isolation
  - Up to 16 per Superdome
  - Independent instance of HP-UX
- Virtual partitions
  - S/W isolation
  - Dynamic cpu allocation
  - Up to 64 per Superdome
  - Independent instance of HP-UX



nPartitions for superdome



Increased system utilization

- partitioning Superdome into physical entities: up to16 nPartitions

Increased Flexibility: Multi OS

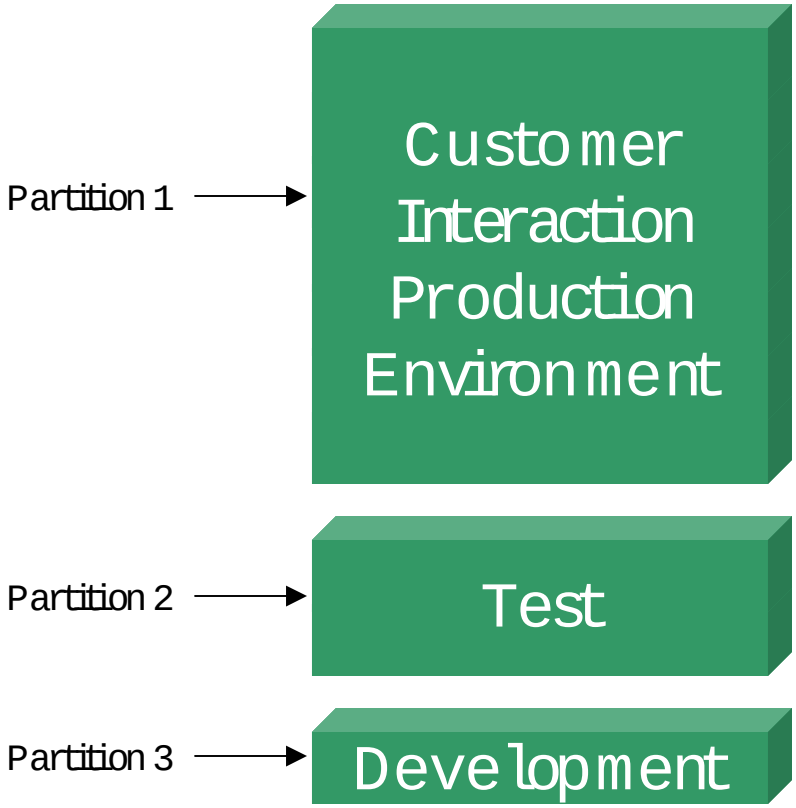
- Multi OS support: HP-UX, Linux (*), Windows (*)
- Multi OS version support
- Multiple patch level support

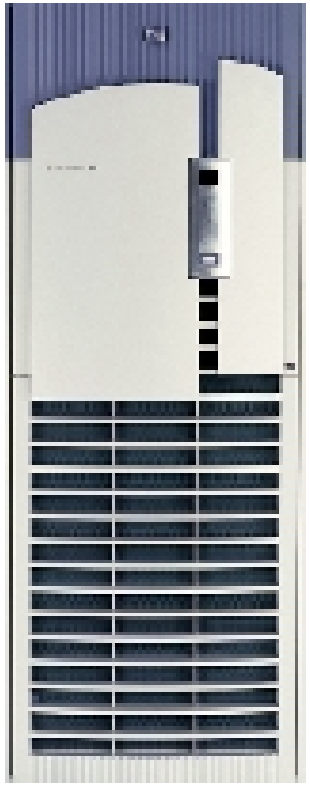
Increased Uptime

- hardware and software isolation across nPartitions
- MC/ServiceGuard support (within Superdome or to another HP 9000 server)

Available in 2H2003

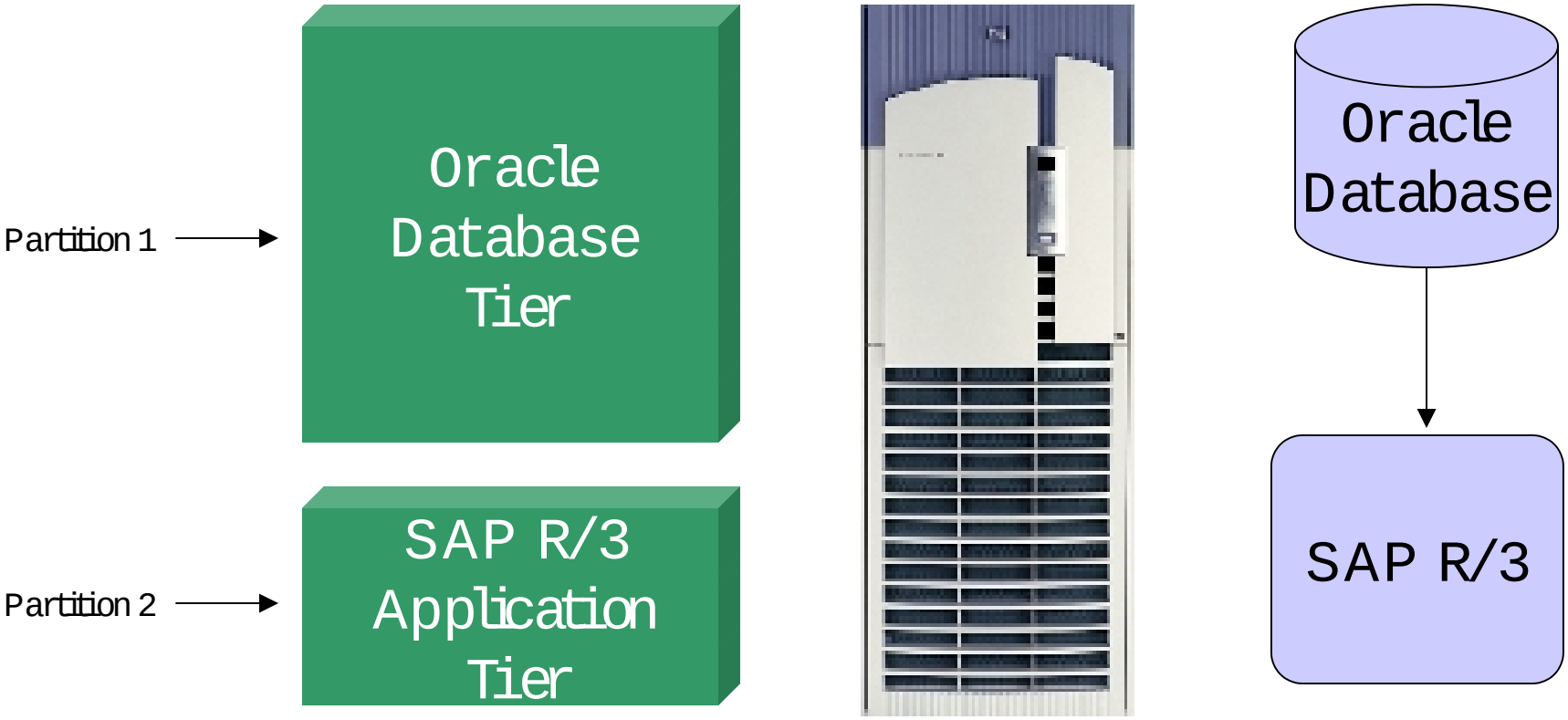
partition examples
isolation of production from test & development



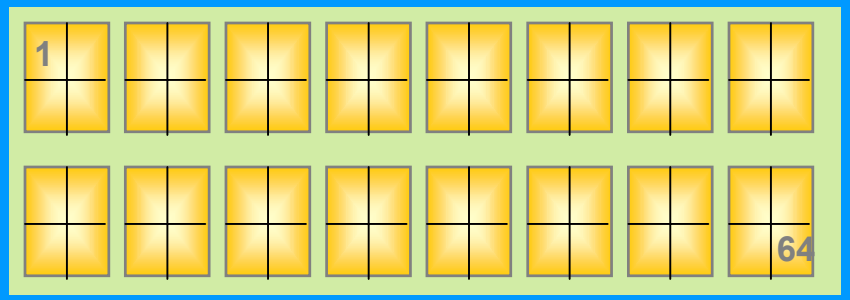


32-way Superdome

partition examples
multi-tier applications



virtual partitions



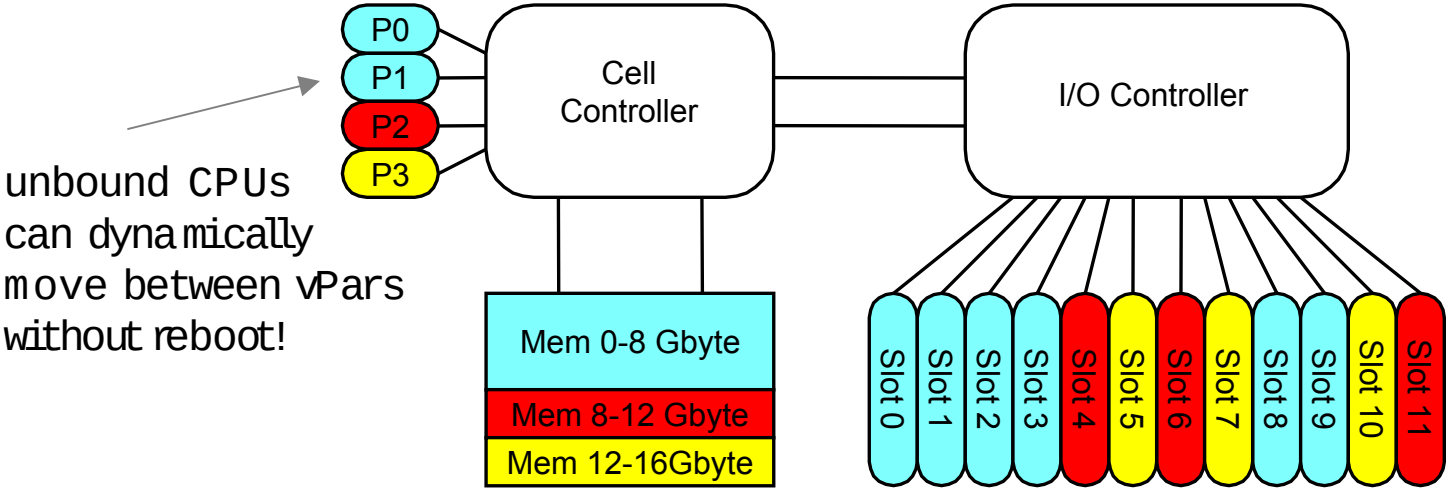
16 Superdome nPartitions can be further partitioned into 64 virtual partitions

- A single nPartition may be soft-partitioned into multiple virtual servers
- Each virtual partition (vPar) runs an independent instance of HP-UX, providing complete name-space isolation
- vPars may run separate release and patch levels of HP-UX
- vPars may be individually reconfigured and rebooted
- Dynamic reconfiguration of CPUs

Available on Superdome now

three virtual partitions on a single cell superdome npartition

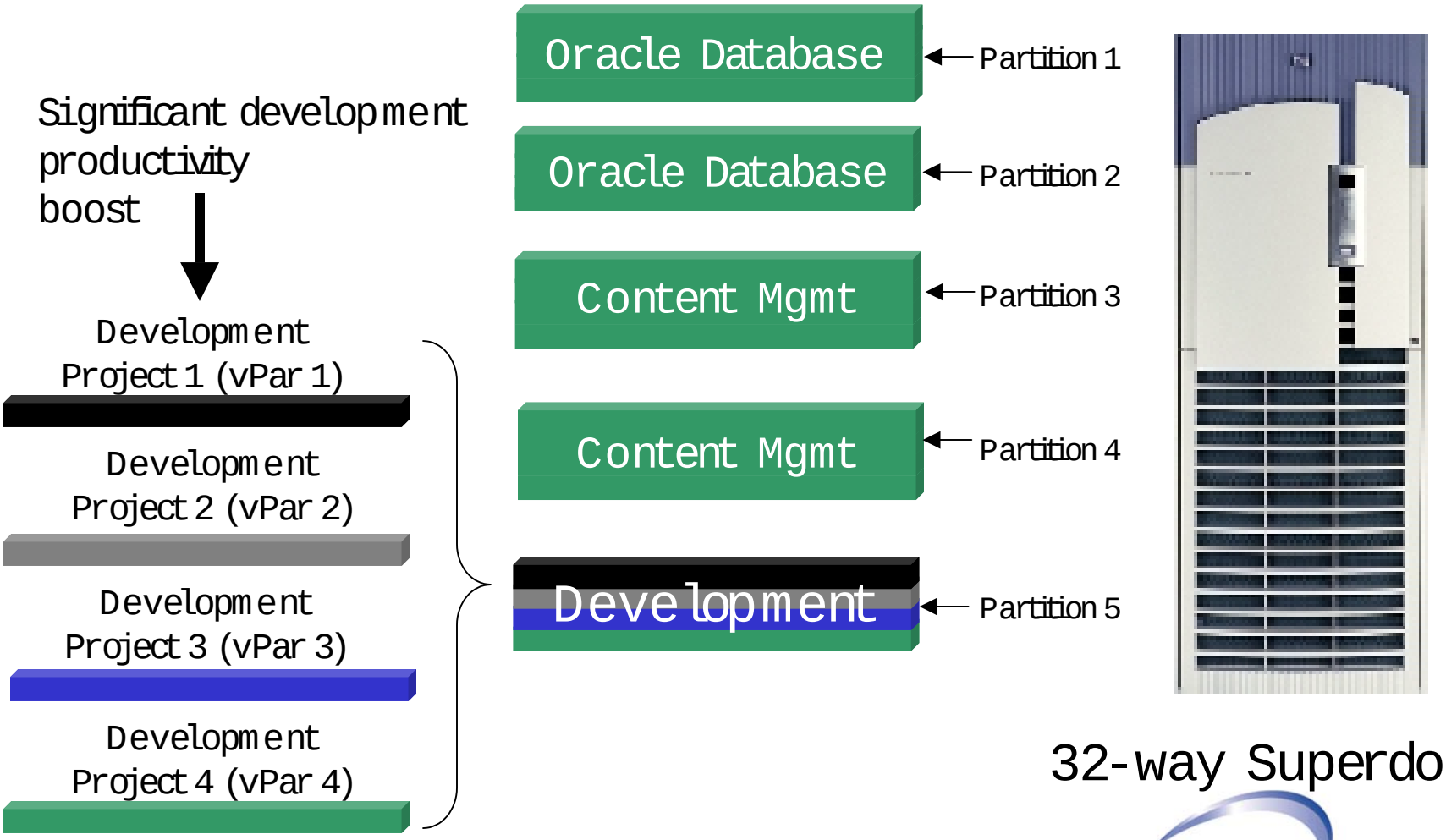
Blue Partition Uses two CPUs, 8 Gbyte memory, 6 PCI slots.
Red partition uses one CPU, 4 GByte memory, 3 PCI slots.
Yellow partition uses one CPU, 4 GByte memory, 3 PCI slots.



Example: CPU P1 (an unbound CPU in this case) can move from blue to red partition when the red partition needs more processing power.

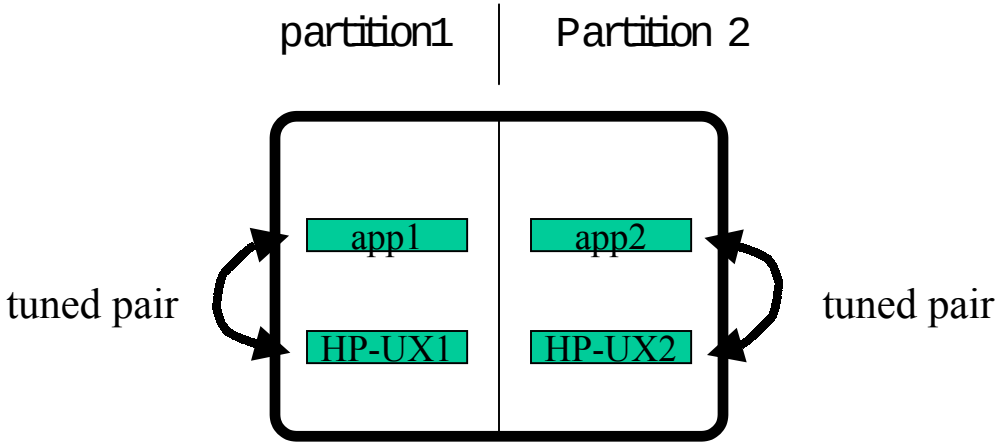
customer example--
vpartitions to isolate development environments

Significant development productivity boost



32-way Superdome

Virtual or hard partitions – a partition performance payoff

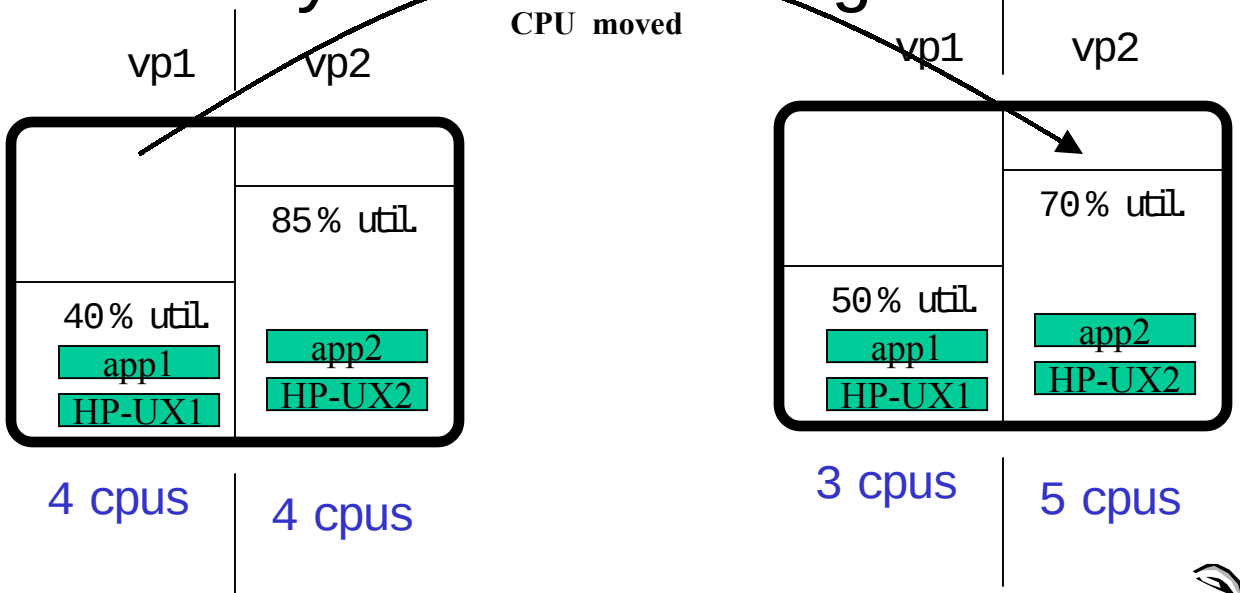


application 1 & HP-UX 1 are optimally tuned for each other

application 2 & HP-UX 2 are optimally tuned for each other

Unix likes to be tuned for a single application

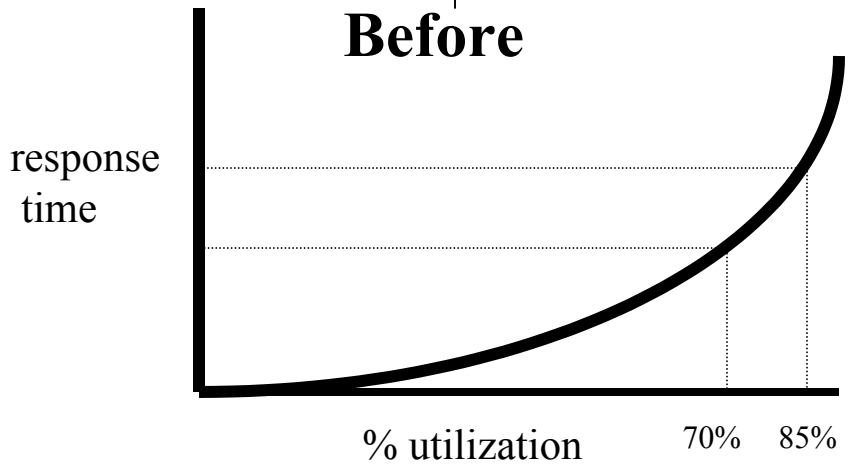
Virtual partitions for increased performance – dynamic CPU migration



Vpars can Cross cell boundaries

Before

After



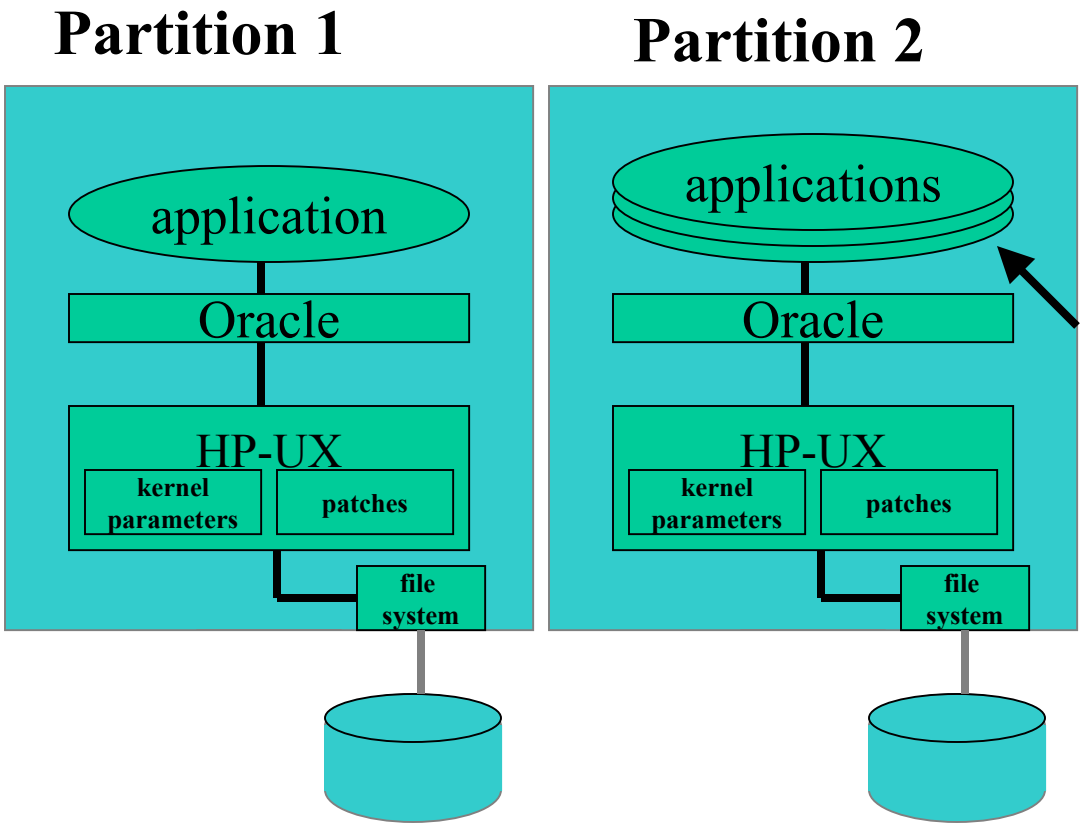
Performance optimization

- use partitions to allow HP-UX to be ideally tuned to the application

-application tuning
-kernel tuning for most common environment
-optimize the file system if I/O intensive

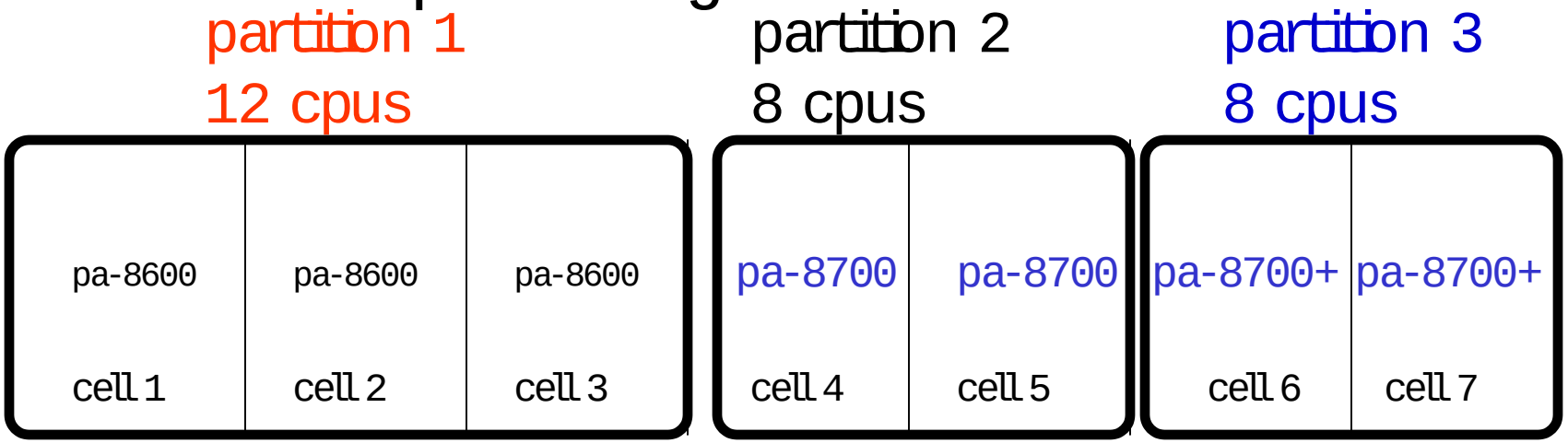
-use patches
- PHCO_26466
- PHKL_26468
for POSIX threads apps for big payoff

Tools:
IOzone
Glance Plus



multiple copies of apps if they don't scale to larger no. of cpus

Performance boost with faster processors while protecting investment



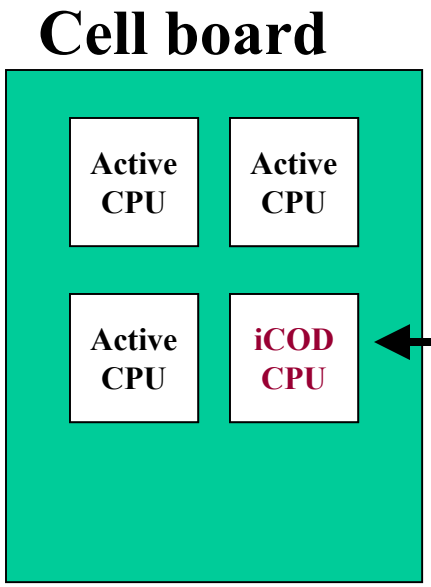
partition 1: keep pa-8600s for investment protection and use this partition for non performance sensitive applications

partition 3: upgrade to pa-8700+ for performance demanding applications

partition 2: use for medium performance sensitive applications

can upgrade to pa-8700+ **on-line, *one partition at a time,*** so applications running in other partitions can keep running.

iCOD (instant capacity on demand)



iCOD standby CPU

activate when needed

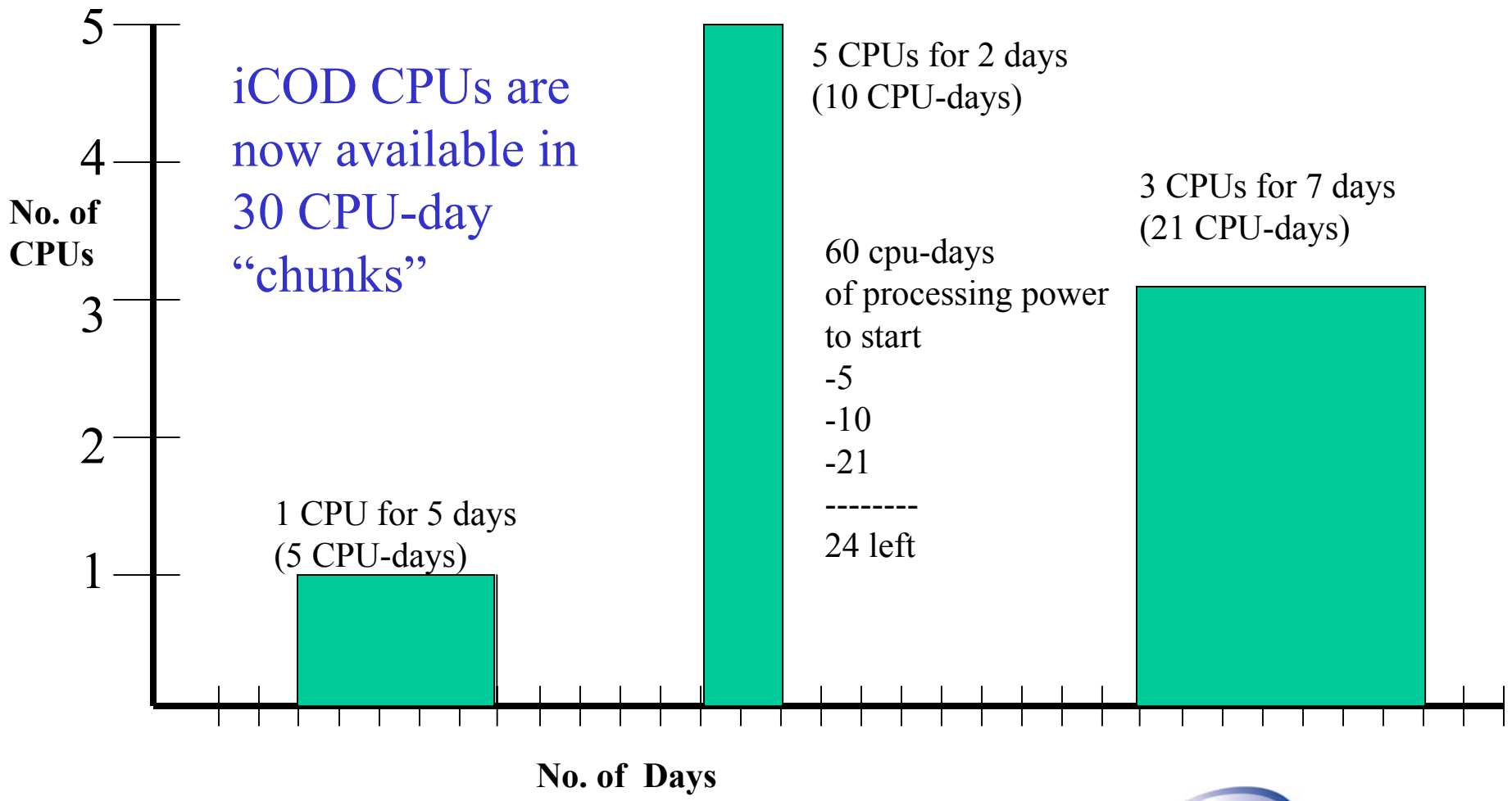
- Two types of iCOD
  - Permanent
  - Temporary capacity

new

iCOD temporary capacity

- provides a performance boost when needed
  - End of month/quarter/year
  - Unexpected peak in demand, activate by command
  - During development stress testing etc. etc.
- Extremely flexible—spread out the “30 CPU-days”
- Moves closer to the “computing utility” concept

add CPUs when needed for performance—iCOD Temporary Capacity

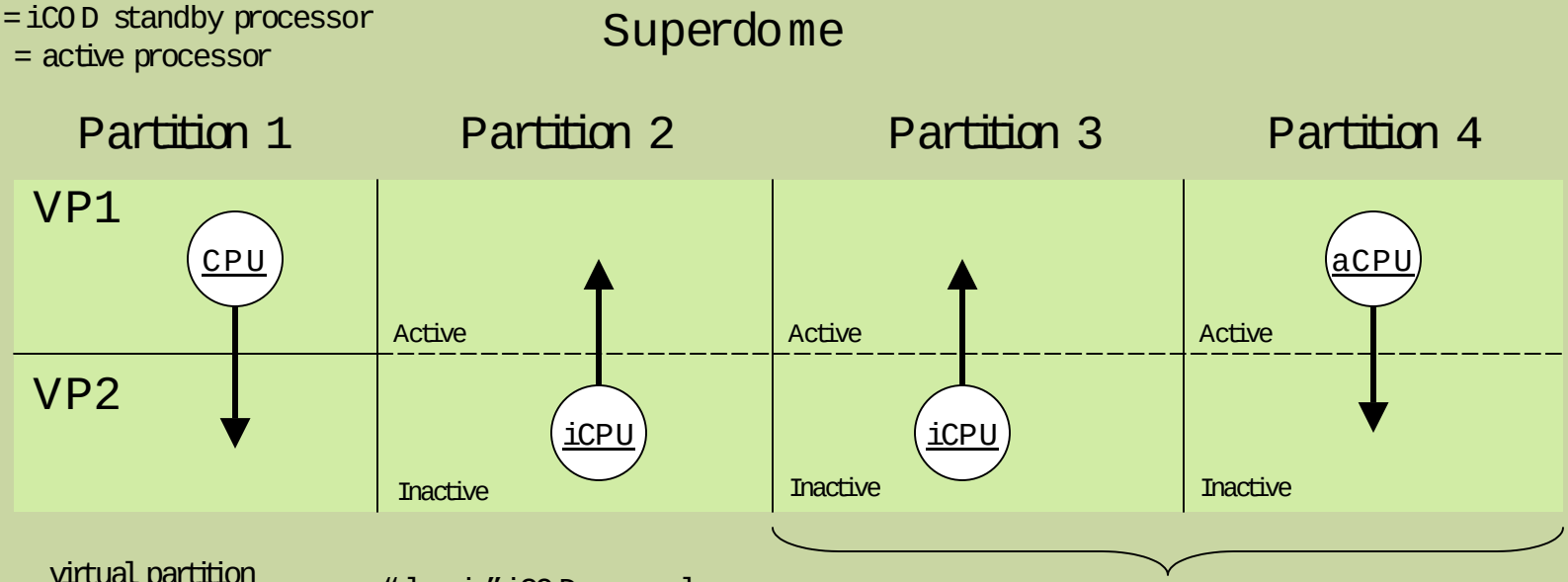


Dynamic CPU Allocation Capabilities

- **Virtual partitions**
  - – Dynamically move unbound CPUs between partitions w/o reboot—by command or automatically via WLM
- **iCOD** (instant capacity on demand)
  - – Dynamically activate a CPU w/o reboot by
    - Simple command . . . or . . .a script . . . or
    - Automatically by goal based WLM when more resources are needed such as to maintain a SLA of 2 second response times
  - – For load balancing, can activate CPU in one hard partition without reboot and deactivate CPU in another hard partition for no charge

Dynamic CPU Allocation Capabilities

VP = virtual partition
iCPU = iCOD standby processor
aCPU = active processor



virtual partition example:

dynamically move unbound CPUs across virtual partitions

(no reboot required)

"classic" iCOD example:

activate iCOD CPU by
1) command or
2) automatically by WLM to meet response time goal (goal based WLM)

(no reboot required)

load balancing example

- activate iCPU processor in one partition (3)
-deactivate aCPU processor in another partition (4)
- no charge

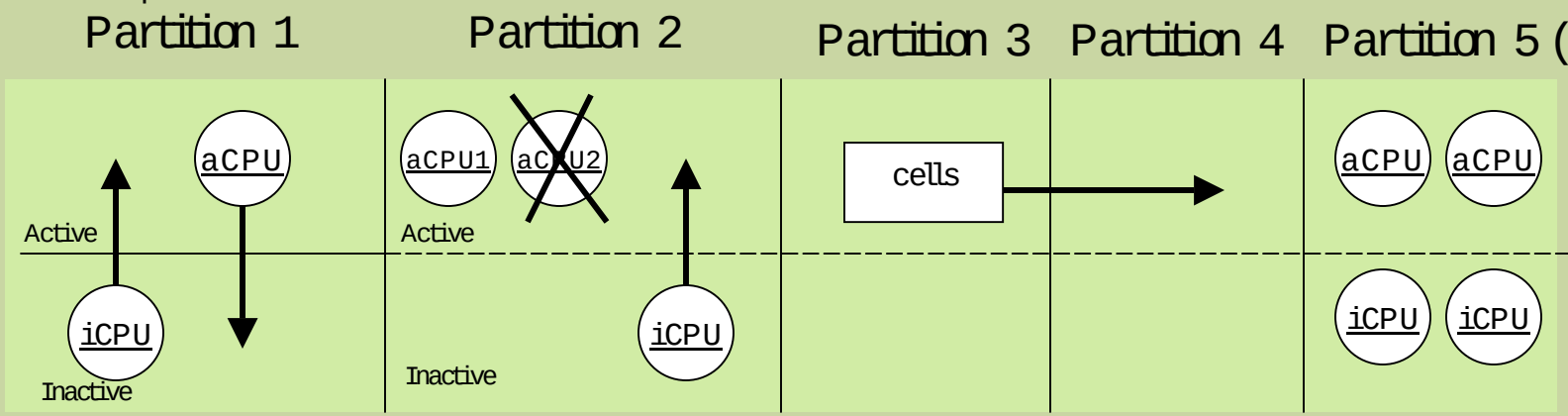
(no reboot required)

Dynamic CPU Allocation Capabilities

- **Utility pricing**
  - Can activate and deactivate CPUs w/o reboot to meet usage demands and save money
- **Deallocation of "misbehaving" CPUs**
  - automatically deactivate one or more "troubled" CPUs and keep the application running w/o reboot. (CPU granularity of 1, not Sun's 4)
    - A: later replace CPU and reboot at a convenient time . . . or . . .
    - B: with iCOD, automatically replace the deactivated CPU immediately with out a charge & without a reboot
- **Hard partitions**
  - Today: Can add CPUs by adding a new hard partition.
  - Today: can remove cells from one partition and add cells to another partition without rebooting any of the non affected partitions (dynamic Npars)
  - Future: can add or delete cells (CPUs) on line (cell on line add & delete)

Dynamic CPU Allocation Capabilities

VP = virtual partition
iCPU = iCOD standby processor
aCPU = active processor



utility pricing example:

end of month: iCPU gets activated for heavy processing loads

middle of month: aCPU gets deactivated

(no reboot required)

CPU replacement example:

- aCPU2 "misbehaves" and is automatically deallocated
- iCPU CPU is automatically activated to replace it
- no charge

(no reboot required)

"dynamic" cell example:

-"dynamically "allocate cells between partitions (Only requires reboot of partitions involved. The rest of Superdome continues to run.)

new partition example:

"dynamically" add CPUs by adding a new partition (requires initial boot of only the new partition)

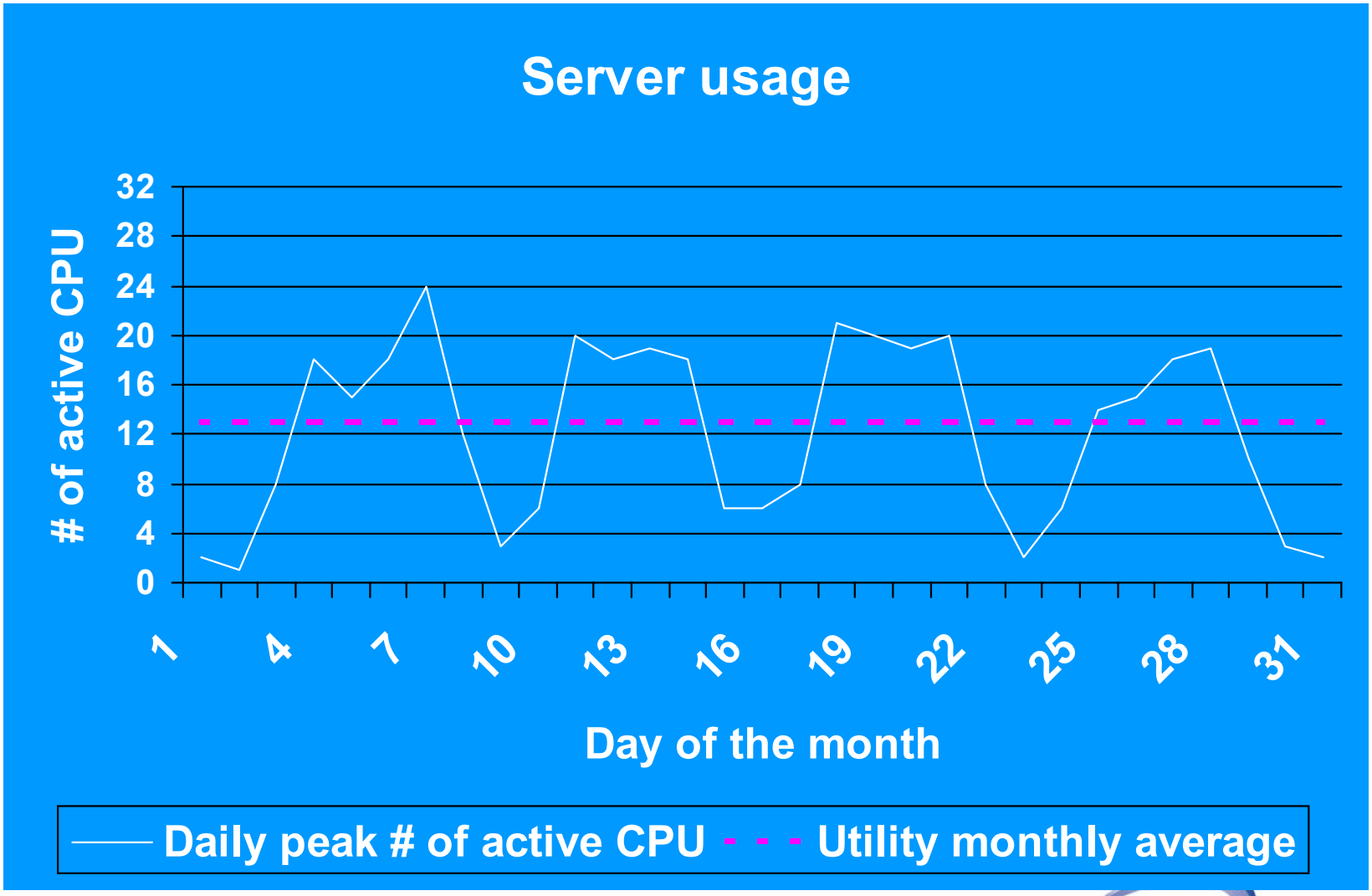
Pay for use (utility concept)

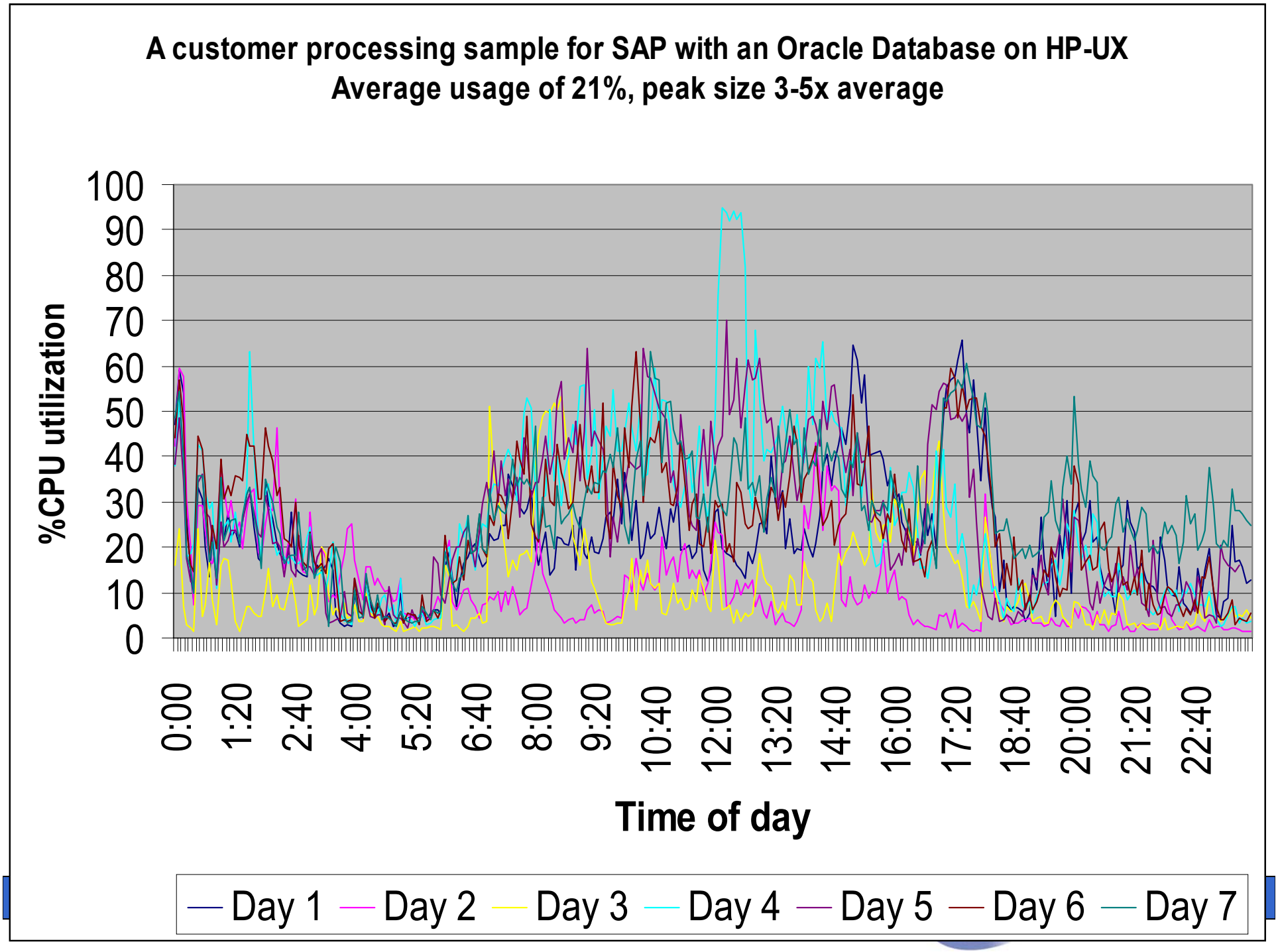
- activate CPUs when needed for performance and deactivate when not needed
- pay based on average no. of CPUs turned on per month
- great for peak periods
  - Tax time
  - End of month/quarter/yr.
  - Election time
  - Holidays
  - Noon/mid morning

helps smooth out IT headaches

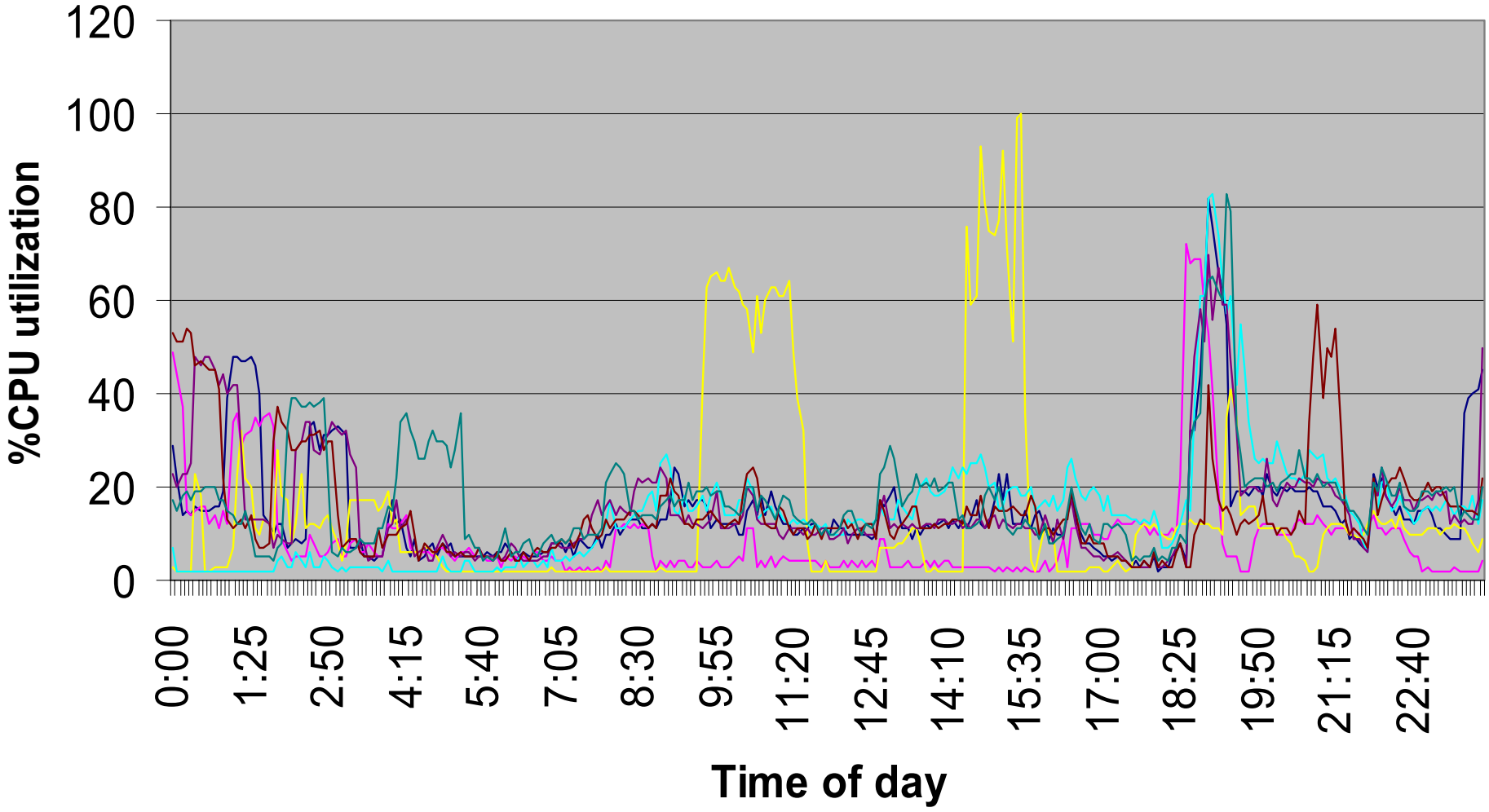
metering server usage

Utility customer's invoice is based on monthly average



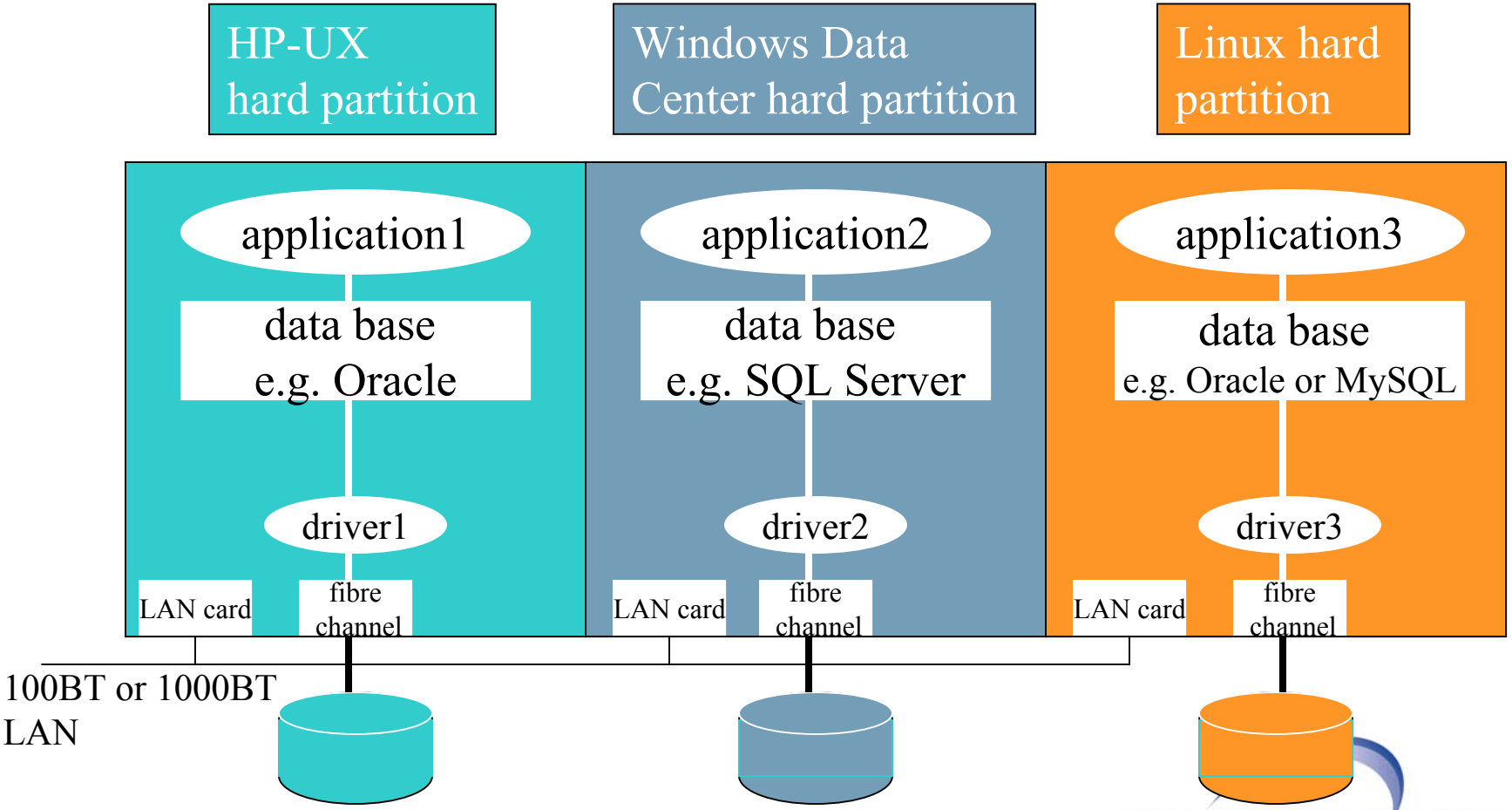

A customer processing sample for SAP with an Oracle Database on HP-UX
Average usage of 21%, peak size 3-5x average
%CPU utilization
100
90
80
70
60
50
40
30
20
10
0
0:00
1:20
2:40
4:00
5:20
6:40
8:00
9:20
10:40
12:00
13:20
14:40
16:00
17:20
18:40
20:00
21:20
22:40
Time of day
Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7

A customer processing sample for a Oracle Database on HP-UX
Average usage of 14%, peak size 4-6x average



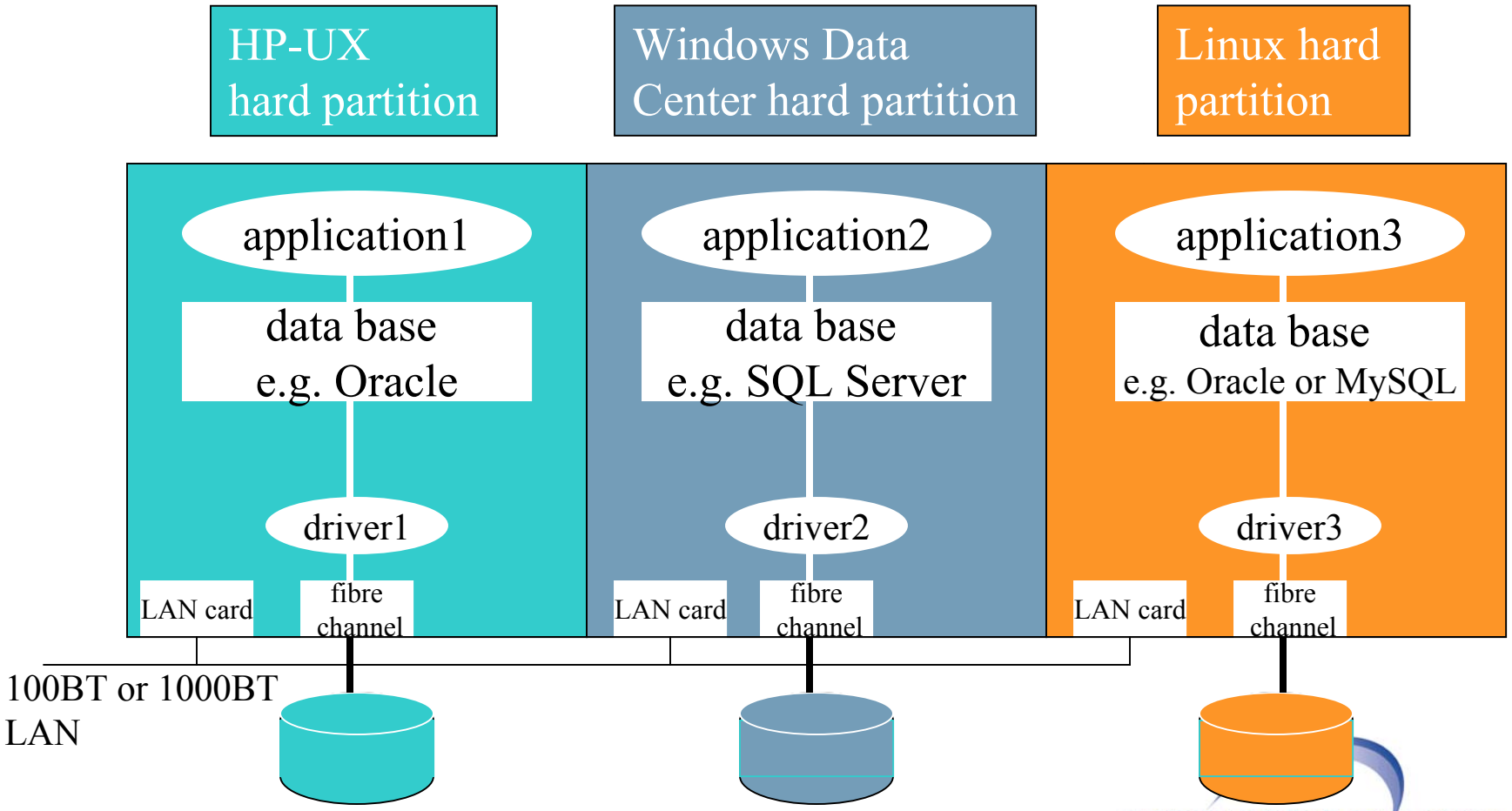
Future Itanium based Superdome (2003)

for performance optimization can pick the best application and operating system combination



Future Itanium based Superdome (2003)

The data base and data base performance most likely will be different.



Summary

- Superdome has many opportunities for increased performance
  - Normal tuning and optimization with only one partition
  - Normal tuning and optimization within each partition
  - Use of partitions in general
  - iCOD addition of cpus
  - Dynamic allocation of cpus
  - Utility cpus on demand
  - Addition of a faster cpu (PA8700+) partition

Attachment—detailed tuning tips

- Goal: these tuning tips are intended to be a short, simplified (not a whole chapter) list of some high payoff tunes.
- Three areas
  - Application tuning
  - Kernel tuning
  - System tuning

Application tuning

- Higher compiler optimization levels are not always the optimum for all applications. Profile (use prof, gprof and Caliper) and use a balanced set of optimization flags for best performance.
- If aggressive optimizations break you applications, identify the offending routine (by binary search) and compile them at lower optimizations.
- Some key compiler and linker flags worth trying out are:
  +O3, +Odataprefetch, +Onolimit, +Olibcalls, +FPD
  (read more about by "man f90", "man cc", "man aCC" etc).
- Build your executable using archived libraries as much as possible. That means link with "-Wl,-archive" or "-Wl,-archive_shared".
- Build and run your applications in 32bit address space unless you have a need to go to 64 bit.
- Set initial data page size to the most appropriate page size using "chatr" command. Read more about by "man chatr".
- For parallel applications, pay special attention to compiler and linker environments to build the most efficient parallel code.

Kernel tuning

- Investigate and optimize the kernel tuning parameters which are most optimum for most of the applications.
- For I/O intensive application, set large buffer cache by a static buffer cache model (set nbuf and bufpagaes to non-zero values) than dynamic model (set dbc_min_pct, dbc_max_pct as % of memory).
- If your runtime environment consists of large I/O intensive processes, set maxfiles, maxfiles_lim and nfile to a large value.
- Make sure the swchunk and maxswapchunks are set large enough to have sufficient swap space. If you cannot for some reason, set swapmem_on=1 to turn on pseudo swap feature.
- For pthread based parallel applications, you may have to increase nkthread parameter appropriately.
- For shared memory parallel applications, set appropriate (high) values for shmmax, shmmni and shmseg.

System tuning

- Install, and periodically update all the OS patches to get the best performance.

- Pay special attention to specific patches for your environment.

  For example, pthread based applications will show significant boost in performance with patches, PHCO_26466 and PHKL_26468.

- Configure your system with enough swapspace. Swap should be more than (or at least equal to) the physical memory. Also, distribute the swap space evenly across drives and controllers.

- For I/O intensive applications, build, test and optimize the file systems with most suitable mount options. Test using applications such as IOZONE or BONNIE.

- Investigate system wide, CPU, Memory and I/O bottlenecks using HP tools such as GlancePlus, TUSC, SAR and VMSTTAT.