Directions with NASDs, SANs, Active Disk and Tapes

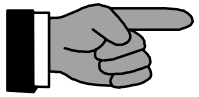
William O'Connell
IBM Toronto Lab

IBM Software

Agenda

- **The Foreseeable Future**
  - ➡ Database Architecture with NASD and SANs

Focus



- **Active Disks, NASD, and SANs**
  - ➡ Directions with active disks, NASD, and SANs
  - ➡ Performance / functionality interfaces enhancements
  - ➡ Practicality of pushing predicate filtering into disks
    - Are we ready for this yet?
- **Tape Impacts on DBMS**
- **Role will optical**
- **Key Points & Challenges**

Disclaimers

- **Will try to stay out of commercial mode, will discuss issues/problems**
- **Even though I/O cost is typically the dominate factor in query processing, it is not our largest problem for the future!!! It is however ...**
  - Ease of use, Administration, and Installation
  - Stability
  - New exotic features (e.g., wizards, ASTs, etc.)
  - Better coordination/collaboration between ISVs and users
- **I am focusing on the directions with database & NASD/SAN interactions**
  - Other panelists will beat on tape needs in future
  - Backup, restore, and log archival is needed for low-, mid-, and high-end systems
  - HSM typically important at high-end only

Basic Question:

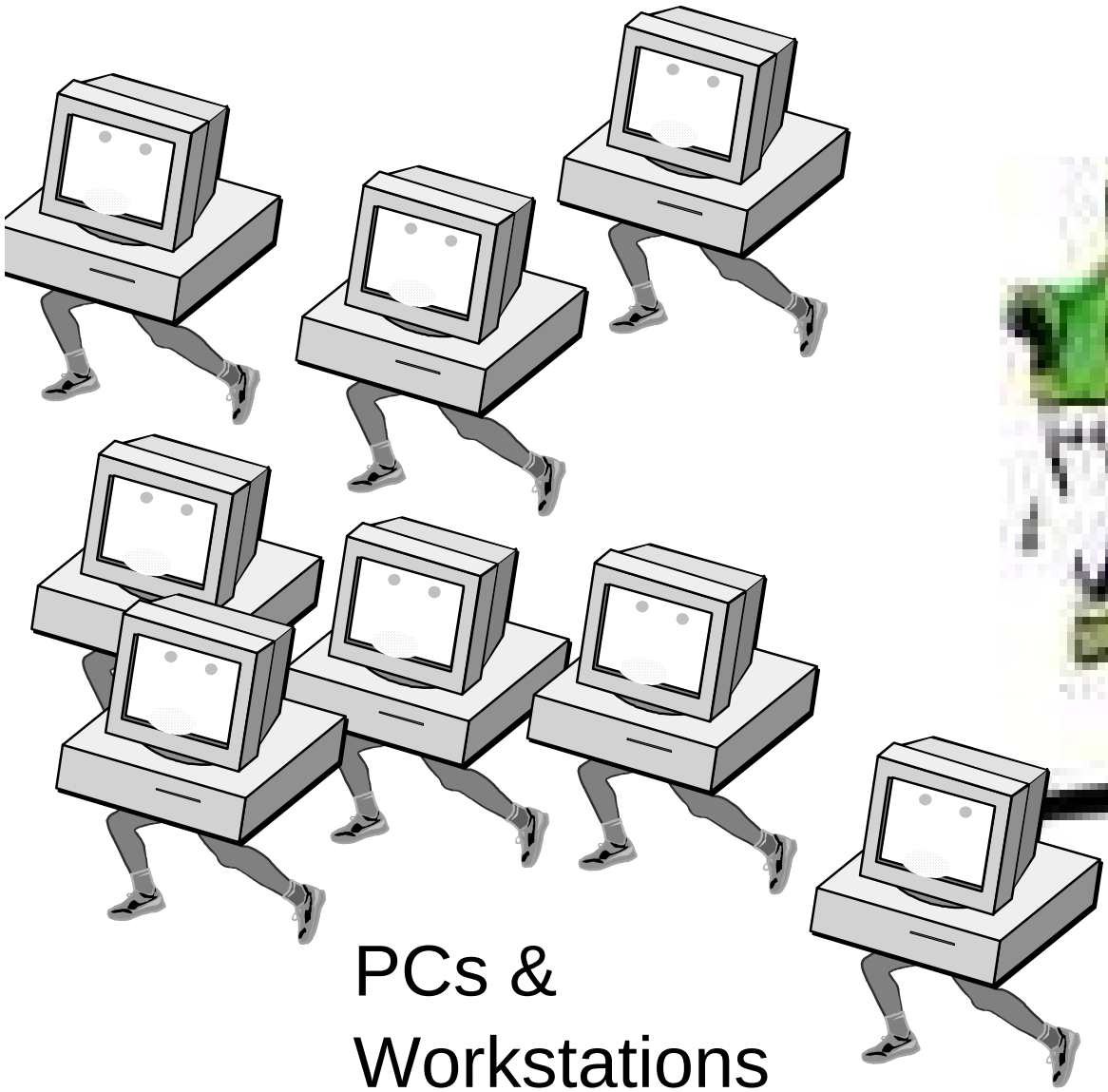
- How can data bases exploit intelligence in storage devices?
- How much functionality do we push into disks/controllers/?
  - Everything? -- Not foreseeable future, not ready yet!!!

Foreseeable Future (synopsis):

- Active (Intelligent) Disk/controllers
  - **Compression of table and/or index data support**
  - **On-line Backup/restore and data versioning support**
  - **Reorg; e.g., write or read affinity on large blocks support**
- NASD & SAN
  - **Opening up data-sharing, e.g., DB2/390 and DB2 Unix/NT share disk subsys (moving data from mainframe to open sys)**
  - **Simplifies data mvmt and sharing (using storage backplane)**

MAINFRAMES, …

We’re back & we’re pissed!!!!!



PCs &
Workstations

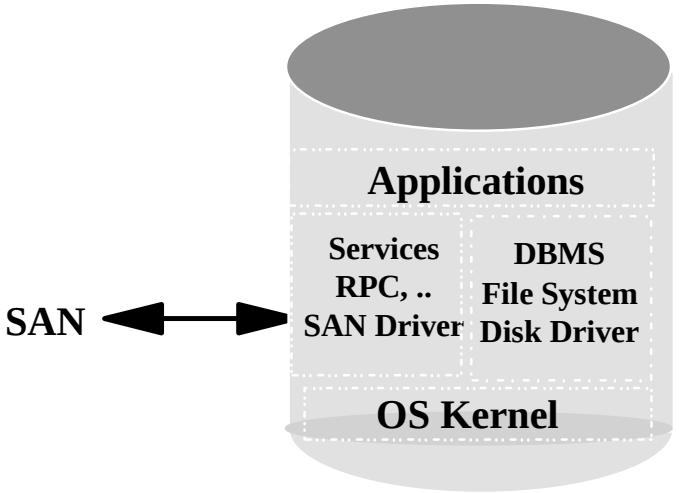
Active Disk, NASD, and SANs (Cont)

- **Disks are becoming a vacuum, similar to Mainframes and Minis**
  - Repeating history, similar to DB2 for MVS/390 and DB2 for AS/400
    - Eventually a vacuum sucks everything up

- **We are not ready to reinvent the database machine quite yet!**
  - We are at the early stages of this vacuum trend
  - Shared everything versus Shared Nothing Issue**s**

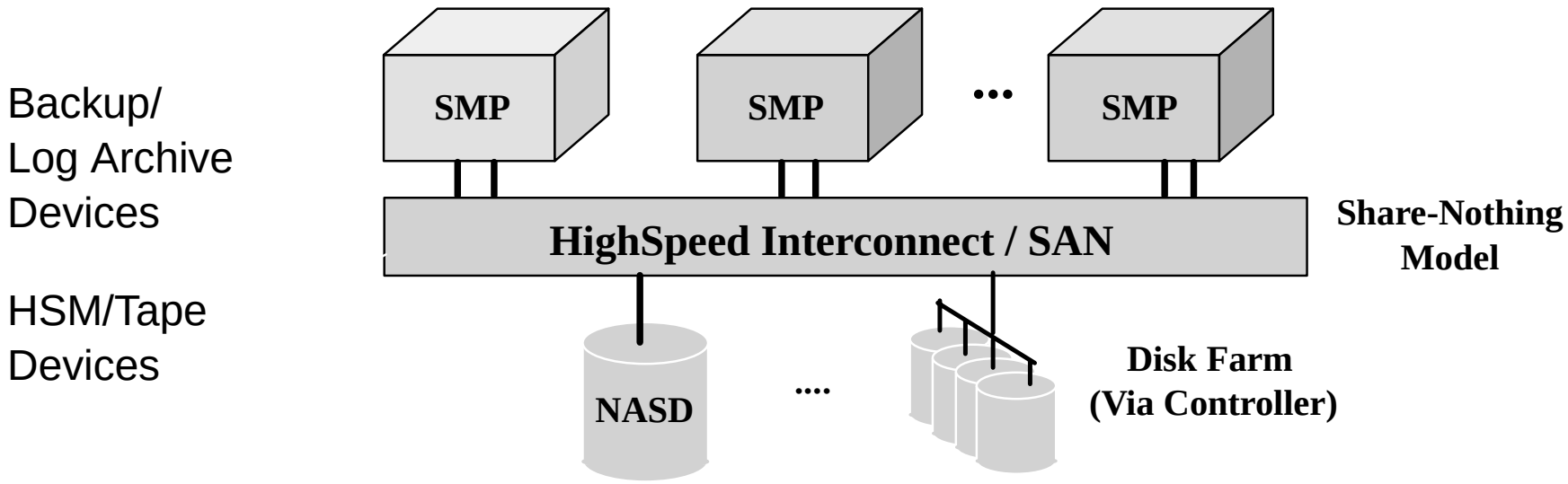
Active Disk, NASD, and SANs (Cont)

- Disk != Database Node (a.k.a., Database Machine)



- Will not happen in Foreseeable Future for Databases

MPP / SMP Hardware Configuration



Foreseeable Future

- Uni -> SMP -> MPP / SMP Scalability
- Configuration planning:
  - Desired cost + Desired Performance + Workload + Data Sizes
  - Many times these contradict, must factor in the importance of each
  - Don't forget about factoring load, backup, and HSM impacts/requirements

IBM Software

Performance / Functionality Interface Enhancements

- **Need larger command list**
  - ➡ Hints (e.g., prevent caching), both in- and Out-of-Band signaling
  - ➡ Hint Examples:
    - Love, Hate, Unknown flags
    - Write-Thru, Write-back caching
      - Optionally guarantee of requested I/O order per session)
- **More outstanding requests (approp for async model)**
  - ➡ Better opportunity for job reordering
    - Want hundreds of outstanding requests
      - Combine N seq I/Os to 1, or dealing w/ 1000s Xcts
    - Allows us to modify/simplify prefetching, scans, etc.
- **Zero copy / Bypass Kernel**
  - ➡ DMA, better L2 cache hit ratios and CPU utilization
    - VI Storage Interface (Intel)
    - Future I/O, NGIO

Performance / Functionality Interface Enhancements (Cont)

- **Remove PCI Bus bottleneck (Largest I/O problem - saturates 1st)**
  - ➡ ATM-Like switch + Channels
  - ➡ Exploit Mainframe Channel host adapter experience
- **Scatter/Gather support for both reads/writes**
  - ➡ For both noncontiguous memory and disk
    - Similar to AIX listio(2)
- **Mismatch between DBMS and I/O controller block mgmt**
  - ➡ Both controller and DBMS manage disk layout
  - ➡ Typically contradict and degrade maximum potential throughout, e.g.,
    - logical volumes mgmt diff; e.g., EMC vs. Symbios
    - DBMSs try to understand differences and optimize (hard)
- **Adaptive block sizes on same disks**
  - ➡ Distinguish between Large vs. small block needs
    - May compete with desire of var-length blks; compression
    - Associated with hints

Performance / Functionality Interface Enhancements (Cont)

- **SAN Network exploitation**
  - ➡ Data sharing of same disk subsystems
    - Does this mean takeover capabilities?
    - Failover common example, but more general
    - Must not forget about locking (all isolation levels!)
  - ➡ Example sharing would be DB2 for NT loading DB2 for MVS/390 data
    - Useful for loading OLTP data into a Warehouse
- **Exploit Log Structured Array (LSA) technology (or similar)**
  - ➡ LSA spts write affinity today (great for tmps, reduce seeks on reads)
    - Minimizes RAID-5 write penalties
    - DB2/390 exploits LSA today
  - ➡ Want read affinity support too!
  - ➡ Exploit following (without copying data, similar to EMC approach):
    - Data Versioning
    - On-line Backup/Restore
    - On-Line Load/Export/Unload

Performance / Functionality Interface Enhancements (Cont)

- **Data Reorganization**
  - ➡ Assist database in reorganization of data
    - Some things most still be done by Database
    - E.g.: record level clustering, record overflows, etc.
- **Support Data Compression on a Volume Basis**
  - ➡ Hardware vs. software assist
  - ➡ For example, DB2/390 has this support today; open system soon
  - ➡ Uncompressed data is moving across network/bus

Should We Push Add'l Query Processing Functionality too?

- **First understand that disks are not the prime bottleneck**
  - PCI is the first thing that saturates
  - Single CPU can push many drives before hitting disk limits
  - We exploit multiple disk arms for parallelism under many wkloads
    - Both SMP parallelism as well as disk arms
- **Help databases by solving the issues already discussed**
  - This is the biggest win between disks and databases now!
- **In future, explore predicate pushdown**
  - However, solving others issues will reduce the need for this
- **Not ready for moving more into Disk/controllers yet**
  - Technology changes quickly, at some point we may be ready

Issues with Predicate Pushdown - Std Interface

Problems that must be overcome!!!

- **Fenced versus Unfenced**
  - Large problem today with UDFs -- DBMSs moving towards PSM & Java
  - Debugging; includes tools + finger pointing!!! Fenced may defeat purpose?
- **Different page formats; same database instance (Legacy)**
  - Most migration is done on the fly,
  - ALTER TABLE COLUMN ADD, etc. etc. (diff per dbms -- need catalog information)
- **Do you follow tombstones or overflow Records?**
  - What about all the system record types? On-line reorg nightmare!!!
- **Locking protocols, especially RR!!! --- Must handle RR, RS, CS, UR Isolations**
  - Rows accessed need to be locked before touched, RIDs are used in lock nm
- **Finding correct row position (double indirection)**
  - Understanding slots, e.g.., empty or delete pending rows (diff per vendor)
  - Some vendor store disk pointer + page base, others multiple this by 2, etc.
- **Complex expression evaluation**
  - E.g., using combos of math, multiple cols, constants, vars (how do this??)

Issues with Predicate Pushdown - Std Interface

(Continued)

- **Same disk, different page size multiples**
  - E.g., 2K, 4K, 8K, 16K, 32K (DB2, ...)
- **Same disk, variable size pages**
  - E.g., 512 bytes - 128Kbs, on sector boundaries (Teradata or compression)
- **Row Compression (DB2/MVS)**
- **Column Compression (Teradata)**
- **Reacting properly to Nulls in row columns**
  - Nulls use bit fields in addition to the column storage (diff per vendor)
- **Different Code Pages**
  - Kanjii, and other double byte character sets
- **Different Collation Sequences**
- **Skipping pages not part of table**

Roles of Tape and HSM devices

- **How databases exploit Tape?**
  - Today integrated into Backup, restore, log archival
  - Must support table migration of active data (HSM)
    - ➡ Query Processing must be able to retrieve (recall) data on tape
    - ➡ Compiler must schedule recalls in advance at run-time
  - DB2 for 390 has native HSM support
  - DB2 for Unix/NT/OS2, HSM options
    - ➡ Table Views span storage types; has optimizer support
      - Transparent to apps; handles rollin/rollout via utility
    - ➡ Table functions
    - ➡ DataJoiner
- **Issues:**
  - HSM and backup/restore must be integrated
  - Simplify Table View DBA interactions (automate)
  - Query Performance on tape (near-line storage)

IBM Software

Optical

- **Is there a role for Optical?**
  - Yes and No
  - DB2 has no plans or needs to integrate further into engine
  - Can be accessed outboard via table function or DataJoiner
  - Optical devices can be used as LOB container containers
- **Issue:**
  - If use for other than LOB data, issue for DBMS metadata

Key Points and Challenges

- **NASD, SANs can be used effectively to solve perf / functionality problems**
  - We need to exploit the DBMS and disk/SAN interfaces better
- **We are not ready to redesign the database machine**
  - We should not push too much query processing into the disks
  - Push things that make sense into disks/controllers (e.g., don't push pred. eval)
- **Tape is becoming a larger issue for query processing integration**
  - Near-line storage needed for occasional queries (e.g., once a year)
    - ➡ Customers are demanding this today!!!
  - Example markets are financial and telco