Networked Data Management Design Points

James Hamilton

JamesRH@microsoft.com

Microsoft SQL Server

SQL Server

Overview

- Changes in the client world
  - How many and what is connected?
  - Is client size and resource consumption the issue?
- Resultant mid-tier & server side implications:
  - Save everything for all time
  - App programming more precious than hardware
  - DB & app admin and training is major deployment barrier
  - Affordable availability in high change systems
  - Redundant data, summary data, and Metadata
  - Data structure does matter
  - Approximate answers quickly
  - Data processing naturally moves towards storage
- Summary

Client Changes: How Many?

- 1998 US WWW users (IDC)
  - US: 51M
  - World wide: 131M
- 2001 estimates:
  - World Wide: 319M users
  - 515M connected devices
- ½ billion based upon conventional device counts

Clients count: Other Device Types

- Connecting TV, VCR, stove, thermostat, microwave, CD players, computers, garage door opener, lights, etc.
- Sony evangelizing IEEE 1394
  - http://www.sel.sony.com/semi/iee1394wp.html
- Microsoft and consortium of others evangelizing Universal Plug and Play
  - www.upnp.org
- On order of billions of client devices

Why Connect These Devices?

- TV guide and auto VCR programming
- CD label info and song list download
- Sharing data and resources
- Set clocks (flashing 12:00 problem)
- Fire and burglar alarms
- Persist thermometer settings
- Feedback and data sharing based systems:
  - Temperature control & power blind interaction
  - Occupancy directed heating and lighting

Device Connect Example: My Home

- Central control of plant watering system
- Central system providing print, file, and www access for all network-attached systems in house
- Central control of 3 sets of aquarium lights
- Remote marine aquarium pump system in garage
- What could be better:
  - Cooperation of lighting, A/C and power blind systems
  - Alarms and remote notification for failures in:
    - Circulations pump
    - Heating & cooling
    - Salinity changes
    - Filtration system
- Many people doing it today: http://www.x10.org

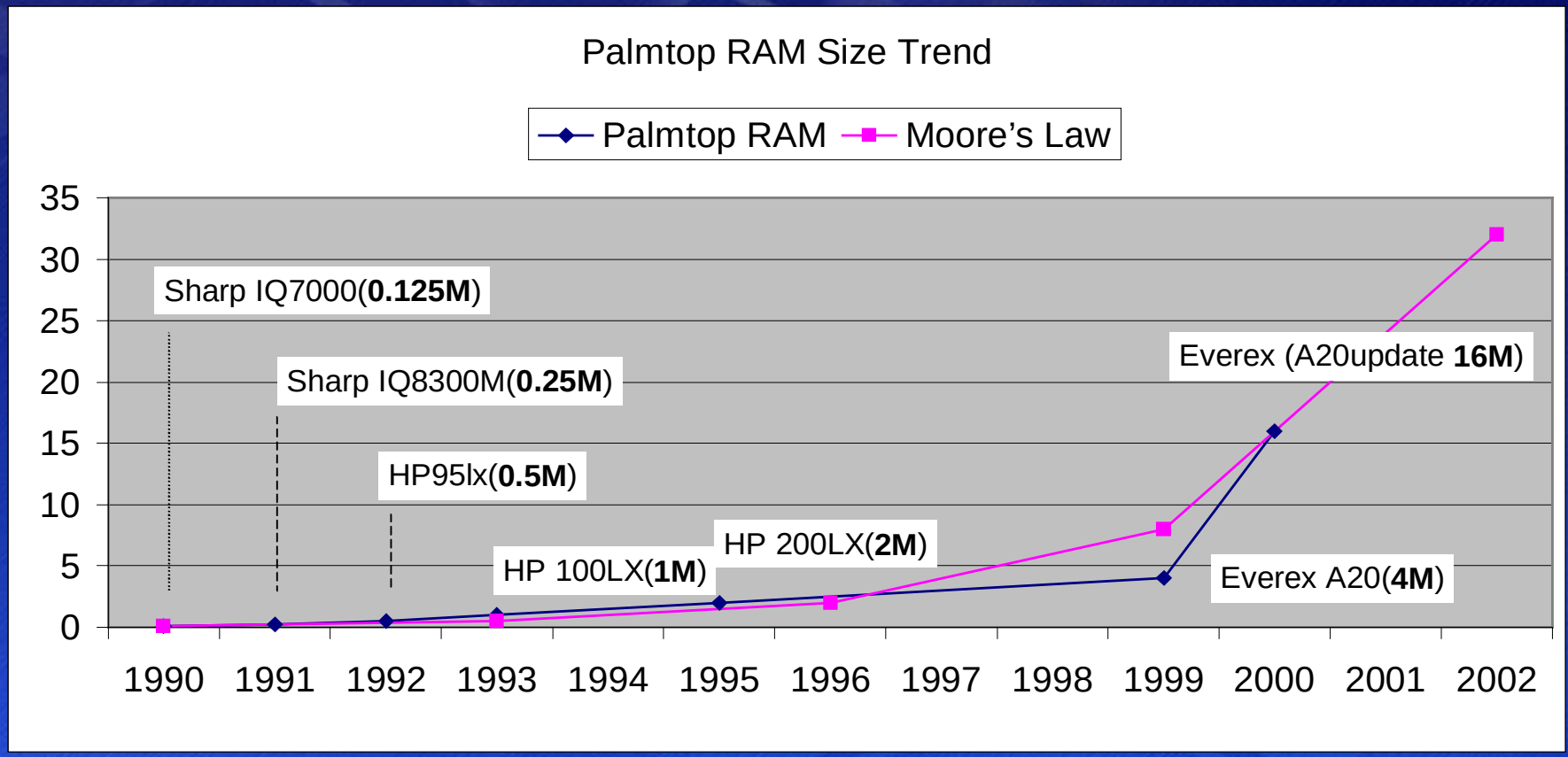
Client Resources the Real Issue?

- "Honey I shrunk the database" (SIGMOD99):
  - Implementation Language
  - DB Footprint
- Both issues either largely irrelevant or soon to be:
  - Dominant costs: admin, operations & user training, and programming
  - Resource availability trends
  - Vertical app slice rather than custom infrastructure

Implementation Language?

- Argument for DB implementation language
  - centers around need to auto-install client side S/W infrastructure (often using Java)
  - Auto-install is absolutely vital, but independent of implementation language
- Auto-install not enough: client should be a cache of recently used S/W and data
  - Full DBMS at client
  - Client-side cache of recently accessed data
  - Optimizer selected access path choice:
    - driven by accuracy & currency requirements
    - balanced against connectivity state & communications costs

Resource Availability Trends



SQLServer

Admin Costs Still Dominate

- 60's large system mentality still prevails:
  - Optimizing use of precious machine resources is a false economy
  - Admin & education costs more important
    - TCO education from the PC world repeated
    - Each app requires admin and user training…much cheaper to roll out 1 infrastructure across multiple form factors
    - Sony PlayStation has 3Mb RAM & Flash
    - Nokia 9000IL phone has 8Mb RAM
- Trending towards 32M palmtop in 2002
  - Vertical app slice resource reqmt can be met

Development Costs Over Memory Costs

- Specialty device & real time O/S typically have weak or non-std dev environments
- Quality & Quantity of apps strongly influenced by:
  - Dev environment quality
  - Availability of trained programmers
- Custom Development & client-side tailoring heavily influence cost & speed of app deployment
- Same apps over wide range of device form factors
- Symmetric client/server execution environment
- General purpose component based DB allows use of required components W/O custom pgming
- DB components and data treated uniformly
  - Both replicated to client as needed

Client Side Summary

- On order of billions connected client devices
  - Bulk are non-conventional computing devices
- All devices include DB components
- Standard physical and logical device interconnect standards will emerge
- DB programming language irrelevant
- Device DB resource consumption an issue but much less important than ease of:
  - Installation
  - Administration
  - Programming
  - Symmetric client/server execution environment

Changes at Mid-tier & Server Side

- All info online and machine accessible
- Redundant data & metadata
- After 30 yrs DB technology more relevant than ever
  - Most people & devices online
  - All devices run DB components
  - Symmetric multi-tier programming model
  - Hierarchical caching model
- Admin including install disappears
- Find structure in weakly/poorly specified schema
- Server availability
- Approximate answers quickly
- Processing moves to storage

Just Save Everything

- Able to store all information produced by our race (Lesk):
  - Paper sources: less than 160 TB
  - Cinema: less than 166 TB
  - Images: 520,000 TB
  - Broadcasting: 80,000 TB
  - Sound: 60 TB
  - Telephony: 4,000,000 TB
- These data yield 5,000 petabytes
- Others estimate upwards of 12,000 petabytes
- World wide storage production in 1998: 13,000 petabytes
- No need to manage deletion of old data
- Most data never accessed by a human
  - access aggregations & statistical analysis, not point fetch
  - More space than data allows for greater redundancy: indexes, materialized views, statistics, & other metadata

Redundant Data & Metadata

- Point access to data, the heart of TP, nearly a solved problem
- TP systems tend to scale with number of users, number of people on planet, or growth of business
  - All trending sub-Moore
- Data analysis systems growing far faster than Moores Law:
  - Greg's law: 2x every 9 to 12 (SIGMOD98—Patterson)
  - Seriously super-Moore implying that no single system can scale sufficiently: clusters are the only solution
- Storage is trending to free with access time prime limiting factor, so detailed statistics will be maintained
- To improve access speed and availability, many redundant copies of data (indexes, materialized views, etc.)
- Async update for stats, indexes, mat views will dominate
  - Data paths choice based upon need currency & accuracy

Affordable Server Availability

- Also need redundant access paths for availability
- Web-enabled direct access model driving high availability requirements:
  - recent high profile failures at eTrade and Charles Schwab
- Web model enabling competition in info access
  - Drives much faster server side software innovation which negative impacts quality
- “Dark machine room” approach requires auto-admin and data redundancy (Inktomi model)
  - 42% of system failures admin error (Gray)
  - Paging admin at 2am to fix problem is dangerous

Server Availability: Heisenbugs

- Industry effective at removing functional errors
- We fail in finding & fixing multi-user & multi-app interactions:
  - Sequences of statistically unlikely events
  - Heisenbugs(research.microsoft.com/~gray/Talks/ISAT_Gray_FT_Avialiability_talk.ppt)
- Testing for these is exponentially expensive
  - Server stack is nearing 100 MLOC
  - Long testing and beta cycles delay software release (typically well over 1 year)
- System size & complexity growth inevitable:
  - Re-try operation (Microsoft Exchange)
  - Re-run operation against redundant data copy (Tandem)
  - Fail fast design approach is robust but only acceptable with redundant access to redundant copies of data

DB Admin Deployment Barrier

- *"You keep explaining to me how I can solve your problems"* (Bank of America)
- Admin costs single largest driver of IT costs
- Admitting we have a problem is first step to a cure:
  - Most commercial DBs now focusing on admin costs
  - SQL Server:
    - Enterprise manager (MMC framework--same as O/S)
    - Integrated security with O/S
    - Index tuning wizard (Surajit Chaudhuri)
    - Auto-statistics creation
    - Auto-file grow/shrink
    - Auto memory resource allocation
- "Install and run" model is near
  - Trades processor resources for admin costs

Interesting Admin-Related Problems

- Multiple cached plans for different parameter marker sub-domains
- Async statistics gathering
- Async optimization
- Feedback-directed techniques:
  - Adapting number of histogram buckets
  - Re-optimizing when cardinality errors discovered during execution
  - re-optimize with additional data distribution info gained during this execution
- Optimizer-created indexing structures:
  - Add indexes when needed (Exchange & AS/400)

Data Structure Matters

- Most internet content is unstructured text
  - restricted to simple Boolean search techniques
- Docs have structure, but not explicit
- Yahoo hand categorizes content
  - indexing limited & human involvement doesn't scale well
- XML is a good mix of simplicity, flexibility, & potential richness
  - Likely to become structure description language of internet
  - DBMSs need to support as first class datatype
- Not enough librarians in world so all information must be self-describing

Approximate Answers Quickly

- DB systems specialize in absolutely correct answer
  - As size grows, correct answer increasingly expensive
- Text search systems: value in quick approx answer
- Approx quickly with statistical confidence bound
  - Steadily improve result over time until user satisfied
- "*Ripple Joins for Online Aggregation*" (Hellerstein—SIGMOD99)
- Allows rapid exploration of hypothesis over very large DB
  - Compute conventional full accuracy report once hypothesis looks correct

Processing moves towards storage

- Trends:
  - I/O bus bandwidth is bottleneck
  - Switched serial networks can support very high bandwidth
  - Processor/memory interface is bottleneck
  - Growing CPU/DRAM perf gap leading to most CPU cycles in stalls
- Combine CPU, serial network, memory, & disk in single package (Patterson)
- Each disk forms a single node of multi-thousand node server cluster
  - Redundant data masks failure (RAID-like approach)
  - Each cyberbrick composed of commodity H/W and commodity S/W (O/S, database, and other server software)
  - Each “slice” plugged in and personality set (e.g. datbase or SAP app server) – no other config
  - On failure of S/W or H/W, redundant nodes pick up workload – replace failures at leisure

Summary

- Order billions of connected client devices
- Client DB footprint and impl lang irrelevant
- Admin costs & prog efficiency are significant issues
- All info online & machine accessible
- Redundant data & metadata
- After 30 years, DB technology more relevant than ever:
  - Most people & devices online
  - All devices run DB components
  - Symmetric multi-tier programming model
  - Hierarchical caching model
- Admin including install disappears
- Discover structure in weakly or poorly specified schema
- Server availability
- Approximate answers quickly
- Processing moves to storage

Networked Data Management Design Points

James Hamilton

JamesRH@microsoft.com

Microsoft SQL Server

SQL Server