Databases Technologies and Distribution Techniques

Dirk Duellmann, CERN

HEPiX, Rome, April 4th 2006

Outline

- A little bit of history
  - Database vendors - RDBMS, ODBMS, ORDBMS, RDBMS, ?
- Database in the Grid
  - HA/Scaling -> Database Clusters (in Luca's talk)
  - Distribution -> Streams
  - Redundancy -> Data Guard
  - Scaling by distribution -> DB Caching Services
- Outlook…

One cycle of HEP use of databases..

- '90: RDBMS + files (DESY, FNAL, CERN,... w/ Oracle)
  - Why? Simpler than older database models!
- '95: ODBMS for "all data" (Babar, COMPASS, HARP w/ Objectivity)
  - Why? Ease of OO language binding!
- '00: Espresso - HEP prototype implementation of ODBMS
  - Why? ODBMS still a nice market...! Need to cotrol the code..
- '01: ORDBMS (Oracle)
  - Why? HEP does not have manpower to write a DB!
- '03: RDBMS + files (COMPASS, HARP, LHC w/ Oracle, MySQL, SQLight)
  - Why? DB vendor abstraction!
    - Assume only commonly available feature set
    - Control object to table mapping inside HEP code (eg POOL)
- More detail -> Jamie's CHEP '06 talk

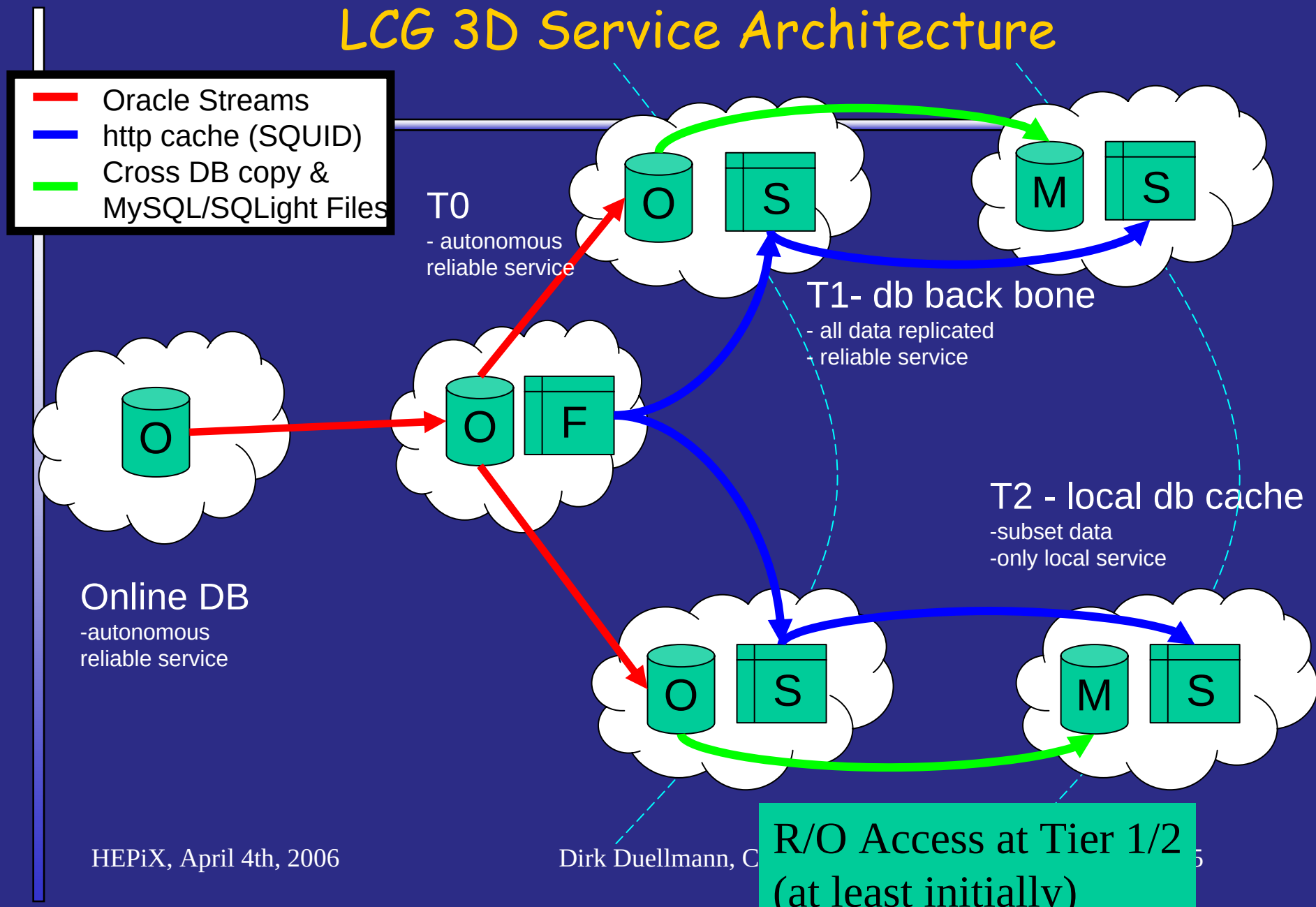
What can/has been learned?

- Changes / decisions were driven by…
  - Changing trust in commercial companies and their long term viability
  - Changing HEP focus (OO application design - rather than reliable service)
- … more than by technology differences

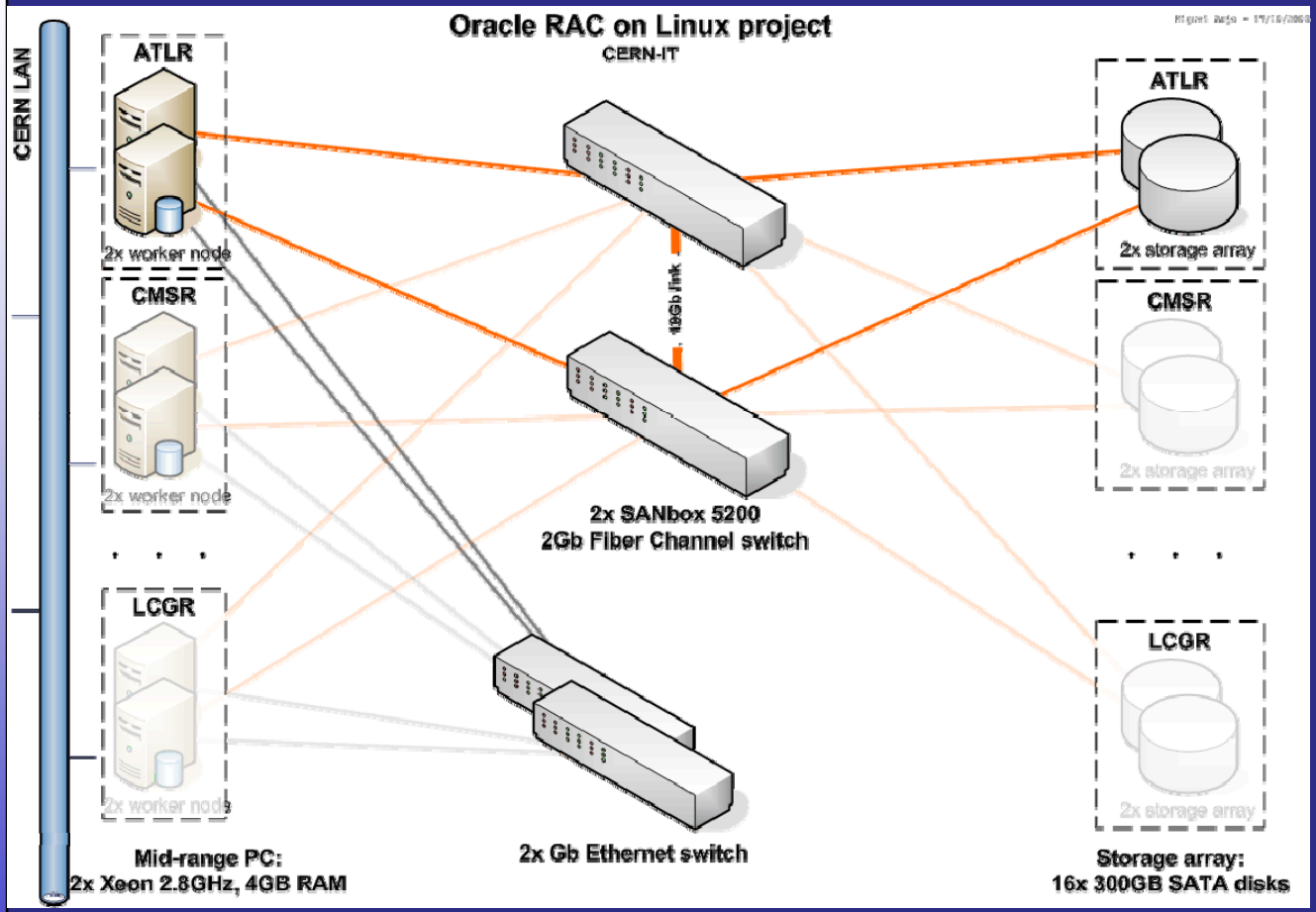
- Several quite different technologies have been used with (some) success
  - Several experiment software frameworks moved along with the changes
  - We learned how to implement proper C++ object bindings (also from ODBMS…)

- Databases host critical data that needs DB features
  - Database technology (still?) too complex/expensive to host more/all HEP data
  - Making DBs simpler/cheaper (by dropping functionality/reliability) does not help

LCG 3D Service Architecture

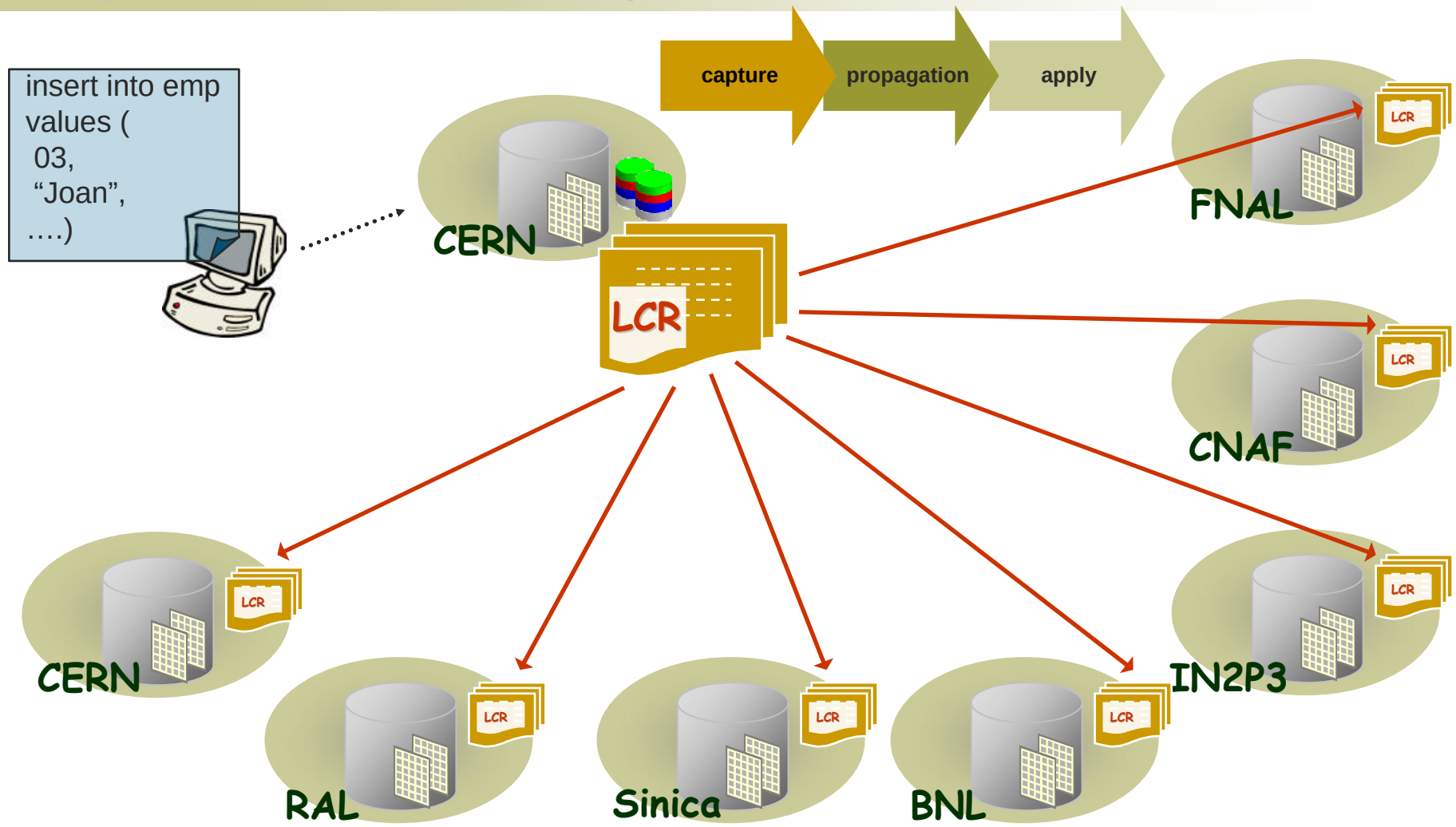


Building Block for Tier 0/1 - Database Clusters (->Luca's Talk)



- Two+ dual-CPU nodes
- Shared storage (eg FC SAN)
- Scale CPU and I/O ops (independently)
- Transparent failover and s/w patches
- LHC database services are deployed on RAC
- All 3D production sites agreed to setup RAC clusters

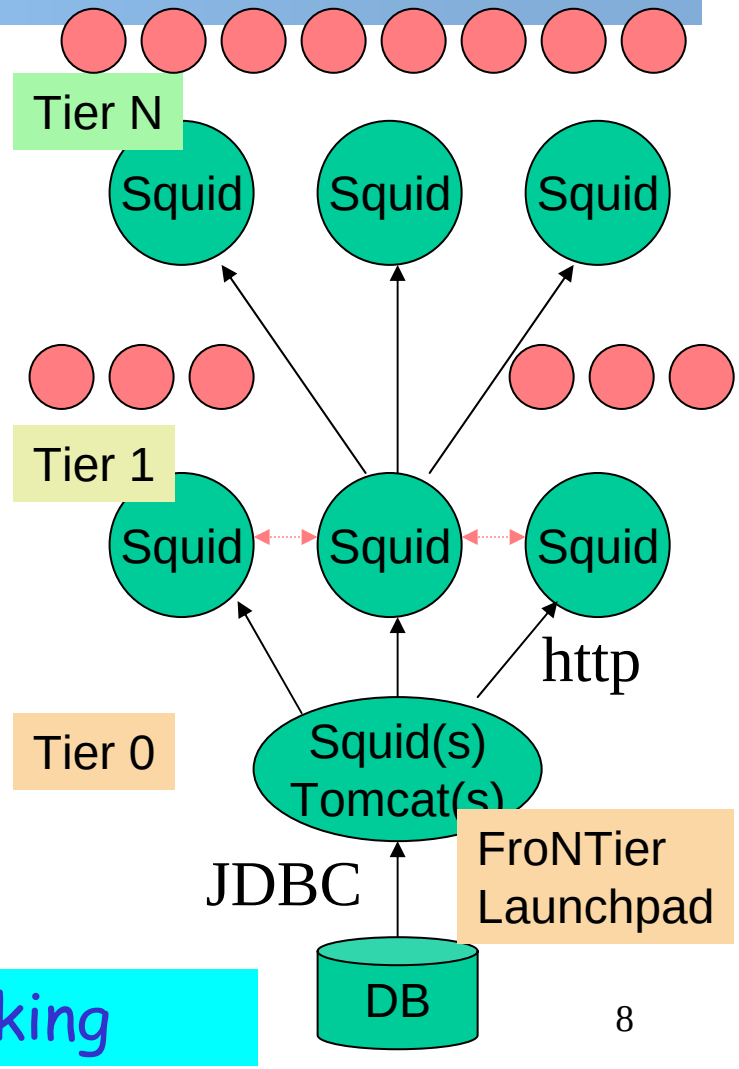
How to keep Databases up-to-date? Asynchronous Replication via Streams



Slide : Eva Dafonte Perez

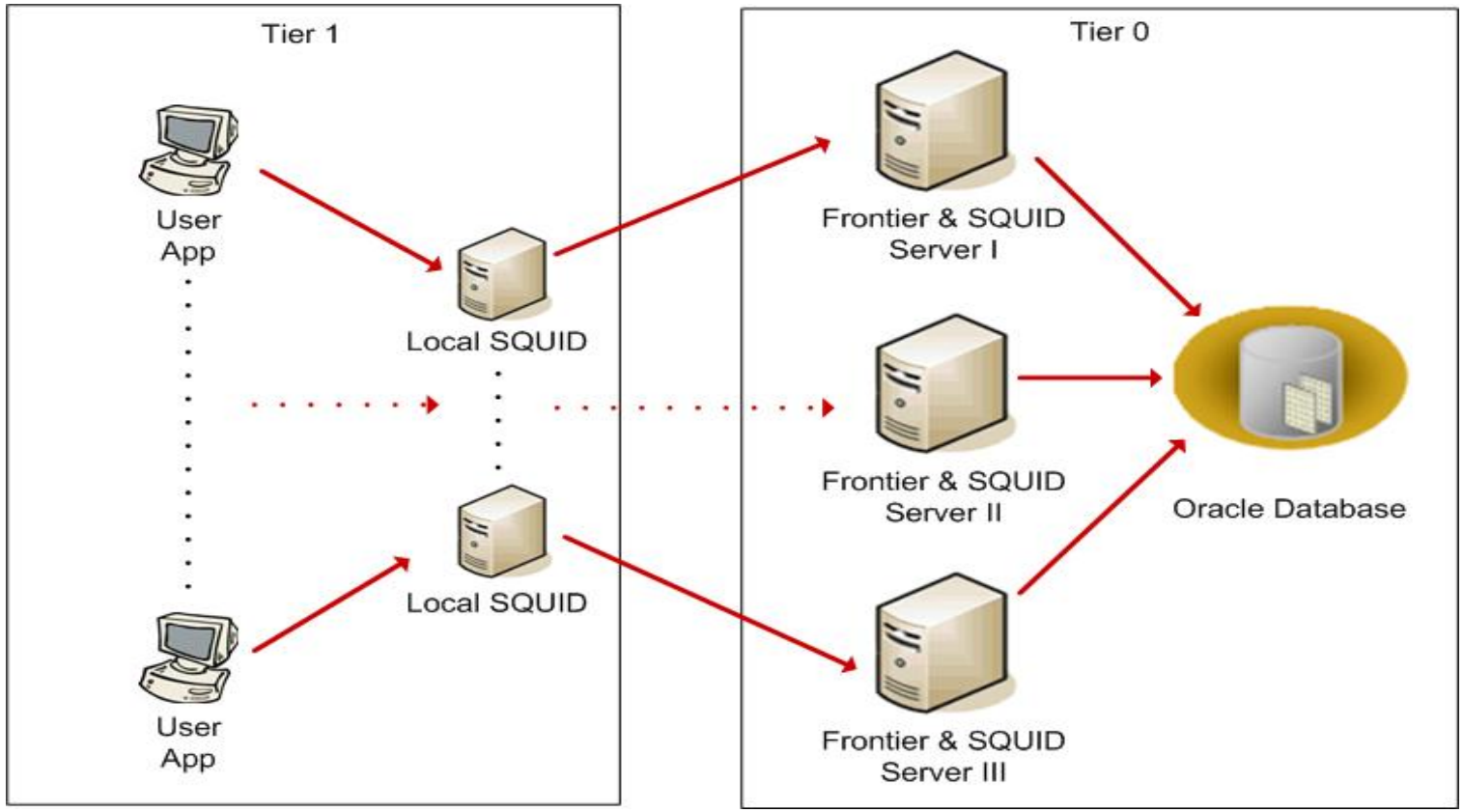
Offline FroNTier Resources/Deployment

- Tier-0: 2-3 Redundant FroNTier servers.
- Tier-1: 2-3 Redundant Squid servers.
- Tier-N: 1-2 Squid Servers.
- Typical Squid server requirements:
  - CPU/MEM/DISK/NIC=1GHz/1 GB/100GB/Gbit
  - Network: visible to Worker LAN (private network) and WAN (internet)
  - Firewall: Two Ports open for URI (FroNTier Launchpad) access and SNMP monitoring (typically 8000 and 3401 respectively)
- Squid non-requirements
  - Special hardware (although high-throughput Disk I/O is good)
  - Cache backup (if disk dies or is corrupted, start from scratch and reload automatically)
- Squid is easy to install and requires little on-going administration.



Slide : Lee Lueking

Frontier Production Configuration at Tier 0



Squid runs in http-accelerator mode (as a reverse proxy server)

Slide : Luis Ramos

LCG Database Deployment Plan

- Two production phases
- April - Sept '06 : partial production service
  - Production service (parallel to existing testbed)
  - H/W requirements defined by experiments/projects
  - Based on Oracle 10gR2
  - Subset of tier 1 sites: ASCC, CERN, BNL, CNAF, GridKA, IN2P3, RAL
- Transfer rate tests schedule with those sites
  - April: complete streams/frontier setup with production DBs
  - May: ramp up to maximum distribution rate on production setup
- October '06- onwards : full production service
  - Adjusted h/w requirements (defined at summer '06 workshop)
  - Other tier 1 sites joined in: PIC, NIKHEF, NDG, TRIUMF

Validation & Throttling

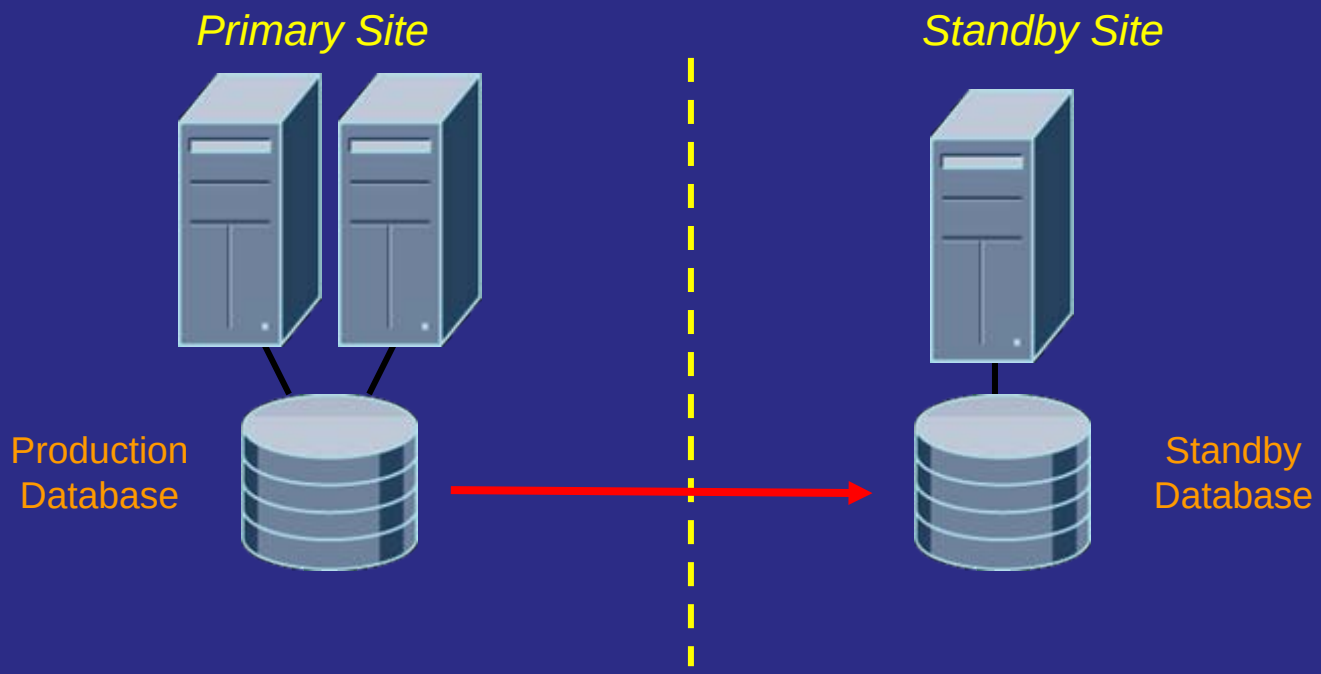
- Different target communities:
  - Application developer: How can I speedup benchmark X?
  - Production mgr: How can I make sure the production uses all resources?
  - DB Admins: How can I make sure the service stays up over night?
- DB Application optimization often perceived as black magic
  - Complex s/w stack with many internal optimizations (& bottlenecks!)
    - CPU, I/O but also net connections, cache use, query plan, table design, indices, bind variables, library cache latch, etc…
- Database react highly nonlinear wrt access pattern changes
  - Need throttling to insure service availability (->Oracle resource profiles)
  - Need developer and DBA together to do validation test at scale

Oracle Enterprise Manager (now: Grid Control)

- Web based user interface
  - Agent based system collecting the status from
    - DB server, OS host, plans for storage & switch plugins
  - DBA level detail and direct changes of DB parameters
    - Customizable reports, metrics and policies
    - Eg: which machines run which DB and OS version and patchlevel
- OEM deployment today
  - Used locally at several HEP sites (eg CERN setup has some 200 targets inside OEM)
    - Evaluating wide area use in LCG 3D testbed
  - New release currently under test
- Very useful as diagnostic tool
  - Need to gain trust to use OEM also as alerting or s/w upgrade tool
  - User access : integration into local fabric monitoring, eg Lemon

Additional Redundancy for Disaster Recovery

- Oracle DataGuard: a copy of the database is kept current by shipping and applying redo logs



Database Futures

- Multicore for DB servers (many threads)
  - HEP setups today use mostly dual CPU cluster nodes
    - R/W applications scaling often limited by cluster interconnect traffic
  - Multicore will allow more CPUs per box (buffer cache)
    - need the memory size and bandwidth to grow accordingly
- 64-bit will allow large server memory
  - more apps to "run in memory" - reduce disk I/O
- Real benefits depend on size of hot data vs server cache
  - Eg size / number of conditions data versions shared by concurrent database clients
  - Needs validation with realistic experiment data models

Conclusions

- The new (and the old) model : RDBMS as part of hybrid approach
  - Object features and vendor abstraction (controlled by HEP code)
  - Databases store key data components in HEP computing (not more yet)
- More recent: Linux DB clusters allow affordable scalable services for LHC startup
  - CERN: 2 sun nodes -> ~50 dual CPU Linux nodes -> ~100 nodes by 2007
  - Recovery of even cheaper IDE DB servers is more expensive
- Grid & DBs require new approaches: WAN connect databases
  - Need to keep their key promises: consistency, reliability
  - Oracle streams and FroNtier under validation
    - Rather complementary than compeeting
- Multicore and 64-bit promise to further reduce disk and IC I/O
- Need larger scale deployment to validate (distributed) DB services
  - Large effort for experiments and sites (while focus is still on files…)