An outside view…

Prof. Eric A. Brewer
UC Berkeley
Intel Research (until July)

Thought about it…

- Most of my wish list hasn't changed much
  - Sigmod 97 keynote about
  - search
  - CIDR 2003 keynote about new areas that don't fit DBMS well
- So, some review, some new stuff

Proposal: Layered Database

- Pros:
  - Enable new database-like things
  - Faster innovation for components
  - Many parallel experiments (like Linux)
  - Should be public domain ideally
- Cons:
  - Hard to ensure global properties
    - But those that care will get them…
- Closest is Berkeley DB (?)

Example: Search Engines

- No use of database technology
- Things that would have been helpful:
  - High availability and replication
  - Atomic version vectors
  - Tools for new declarative languages
  - Join machinery
- Not needed:
  - Complex locks, Query Optimization
  - Transactions, Redo, Undo

Example: Scientific Computing

- Uses databases, but not a good fit
  - Data often stored in files
  - Most operators are outside the DBMS
  - Database is an expensive replicated file system (in/out but no joins)
- Things that layered system might provide:
  - Multi-version storage system
  - New operators
  - Tools for new declarative languages

Other Misfits

- Bioinformatics:
  - Wrong operators
  - Need error propagation
  - Versioning, read mostly
- App Servers:
  - Session state, session migration
  - App server will be a small database
- Workflow

So what happened?

- Accepted: one size does not fit all…
- Couldn't get much traction on layered database
- Built our own from scratch
  - Stasis, Rusty Sears
  - Open source, could be something special
- But big picture largely unchanged
  - Too hard to explore the fun spaces
  - But layering DID happen!
    - But whole database is now just transactional storage

Directions I'd like to see…

- Integrated notion of statistics
  - Store the noise (rather than clean it)
  - Create cleaner views
  - Core probabilistic queries
- Move away from update-in-place
  - Many inputs are sacred (e.g. science)
  - Transactional versioning
  - Provenance & annotation

Directions (2)

- Better integration into PL
- BASE semantics (not just ACID)
- Repeated automatic extraction
  - Web crawlers do this
  - Much of MapReduce workload
  - Need to integrate with versioning, provenance, statistics
  - Import is a continuous process, not an event

Many Core

- Hard to get any performance benefit for I/O bound applications
- Main memory DB??
  - Limited by off-chip bandwidth
  - Need dataflow optimizations on/off chip

Backup

1) Layering enables competition

- Examples from OS community:
  - X86, SPEC benchmarks, Virtual machines
  - SCSI disks, RAID, NAS
  - Routers, Firewalls, Proxies
- Some layers commodities (raw disks)
- Some layers innovative (replication)
- Always have unexpected uses

2) Many more experiments

- Centralized planning tries very few things
- Layering enables many more bets
  - Also enables VC funding
    - Ex: IP layer, ASICs => networking startups
  - Enables niche markets (lower cost of entry)
    - Easier path for XML, bio, spatial, ….
  - Most bets fail, but some succeed

3) Reduces Time to Market

- Lower cost of entry
- More important:
  - Just good enough!
  - Few global properties in early versions
    - The web, search engines, even e-commerce
    - P2P
    - WebMethods
  - Global properties added over time!
- Ugly but fast wins the race…

Claims

- **If you can't control, then enable**
  - This is *the lesson* from OS work for CIDR
  - Unix, TCP *enabled* the web
    - Neither attempted to control usage
  - HTTP in turn *enabled* P2P
- DB research suffers from "Albatross 9i"
  - Artifact hides the enabling technology
  - CIDR exists for this reason

Conclusions

- Can't control (or predict) the future… better to enable broad innovation
- Control
  - Make global properties tractable
  - But limits innovation
- A public domain layered database:
  - Would enable more innovation
  - Allow a broader range of properties

Rate of Innovation

- Claim: layering increases innovation

1) Enables competition
2) Many more experiments
3) Reduces time to market