Hadoop in 15 minutes
Short presentation: Ask Q's.

Brian Bockelman

OSG All Hands Meeting 2009

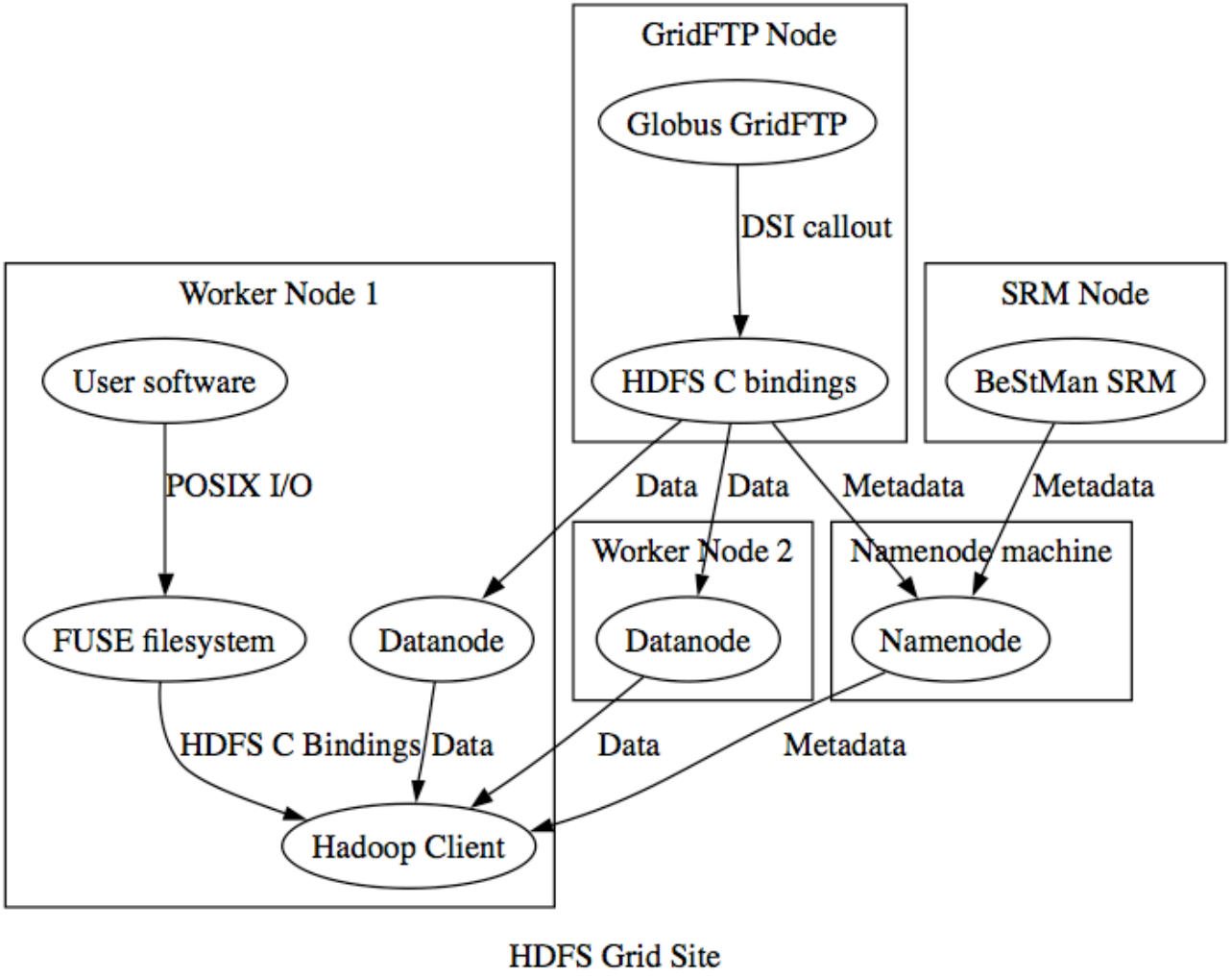
Hadoop

- Hadoop is a data processing system that is an open source implementation of the MapReduce paradigm
- Largest user is Yahoo!; largest cluster is 14PB of disk and 32,000 cores.
- We don't use the data processing parts (yet), but the distributed file system that forms part of the core (HDFS)

HDFS

- 2 major kinds of HDFS nodes:
  - Namenode
  - Datanode
- 2 grid nodes:
  - SRM
  - GridFTP
- 2 auxiliary services:
  - Checkpoint server
  - Balancer

HDFS Diagram

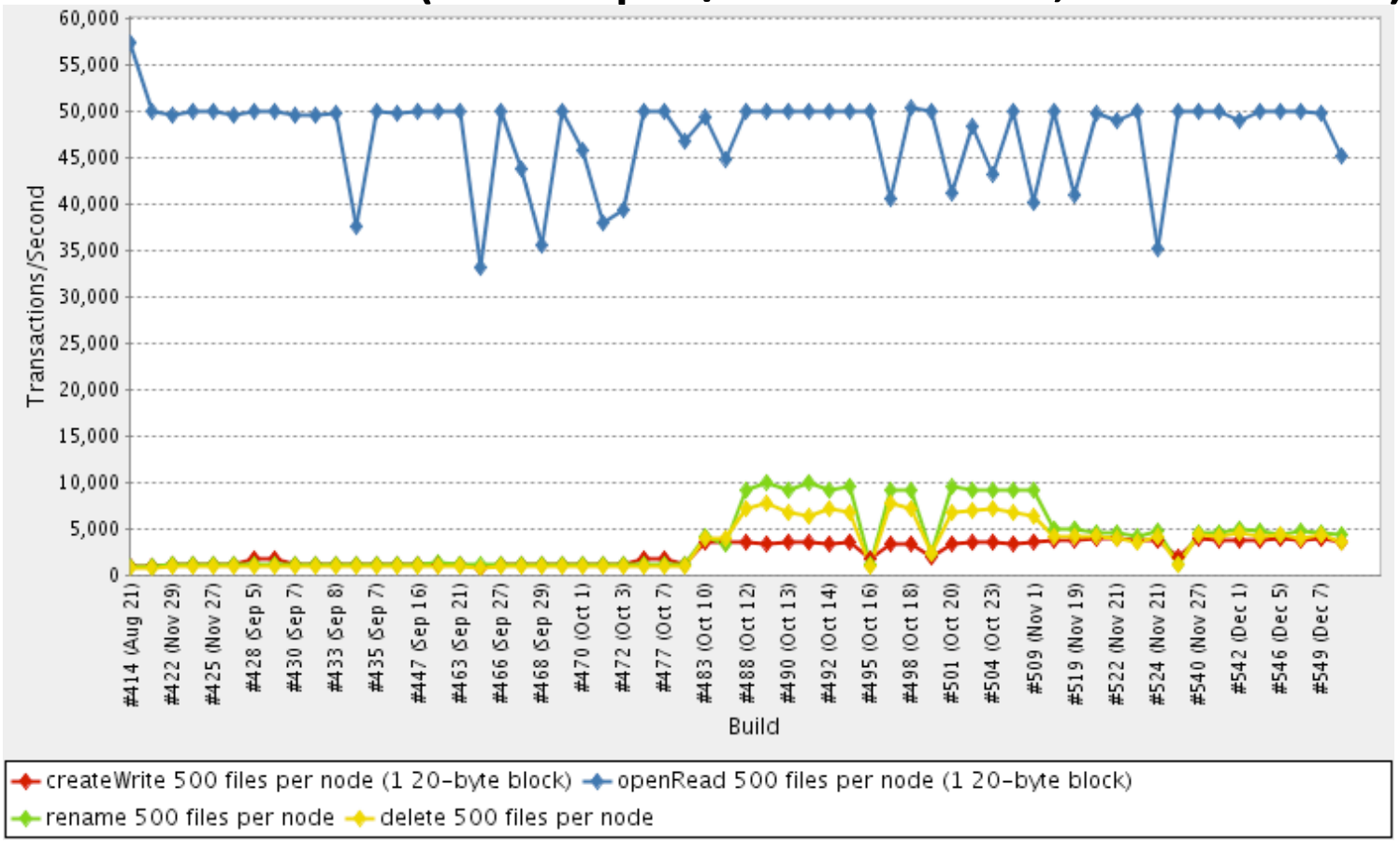


HDFS Details

- Datanode is what you'd expect from a datanode; not much conceptually new.
- Namenode is interesting design.
  - Only namespace and file -> block mapping is persisted.
  - Everything else stays in memory.
    - All operations that do not change namespace only hit RAM; NEVER will hit disk
    - Needs about 1 GB RAM per 1 million files.

HDFS Details

- The Yahoo! folks benchmark the namenode at fantastic rates (50k ops / sec reads, 5k writes)



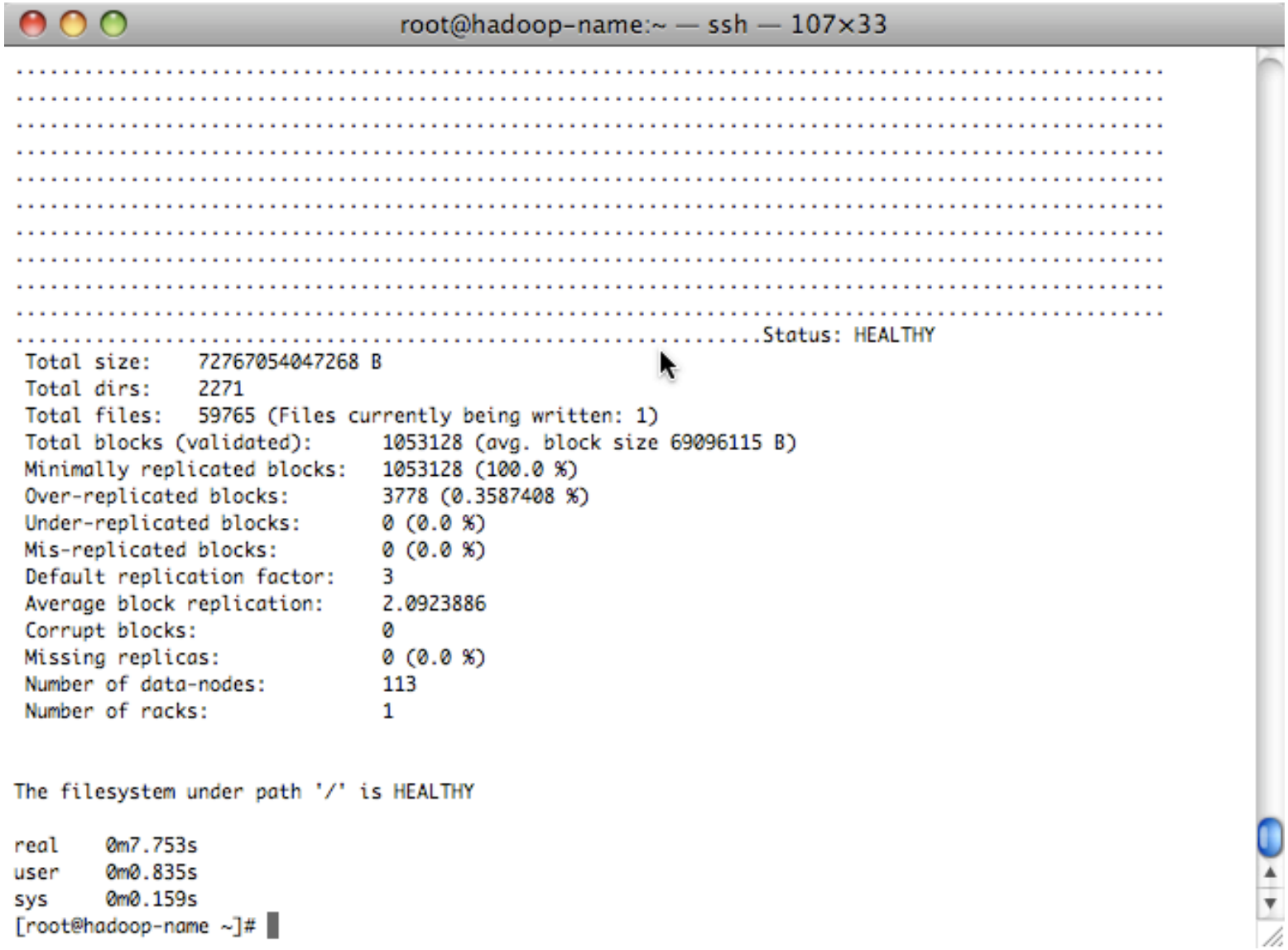
Why HDFS?

- Why chose HDFS? Roughly,
  - Manageability
  - Reliability
  - Usability
  - Performance

Manageability

- The following tasks are trivial:
  - Integration of statistics with Ganglia.
  - Decommissioning hardware.
  - Recovery from hardware failure.
  - Fsck!
    - Checks the current knowledge of the filesystem and counts how many block replicas there are per file, and highlights any which are under-replicated.
  - Pacman-based install for the whole kit.

FSCK example

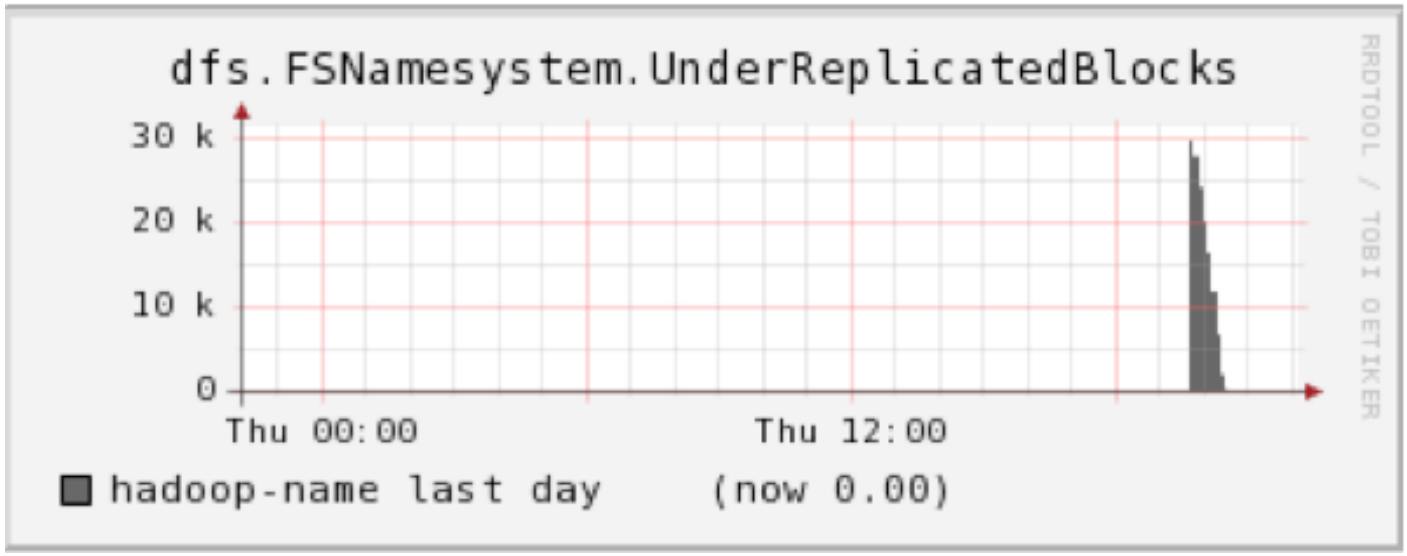


Reliability

- REPLICATION WORKS
  - You can set default replication factors and per-file replication; we have cms production files @ 4 reps
- Client I/O can live through
  - Namenode failures / restarts
  - Datanode failures / restarts
- Semantics guarantee X complete copies when you finish writing each block.

Reliability

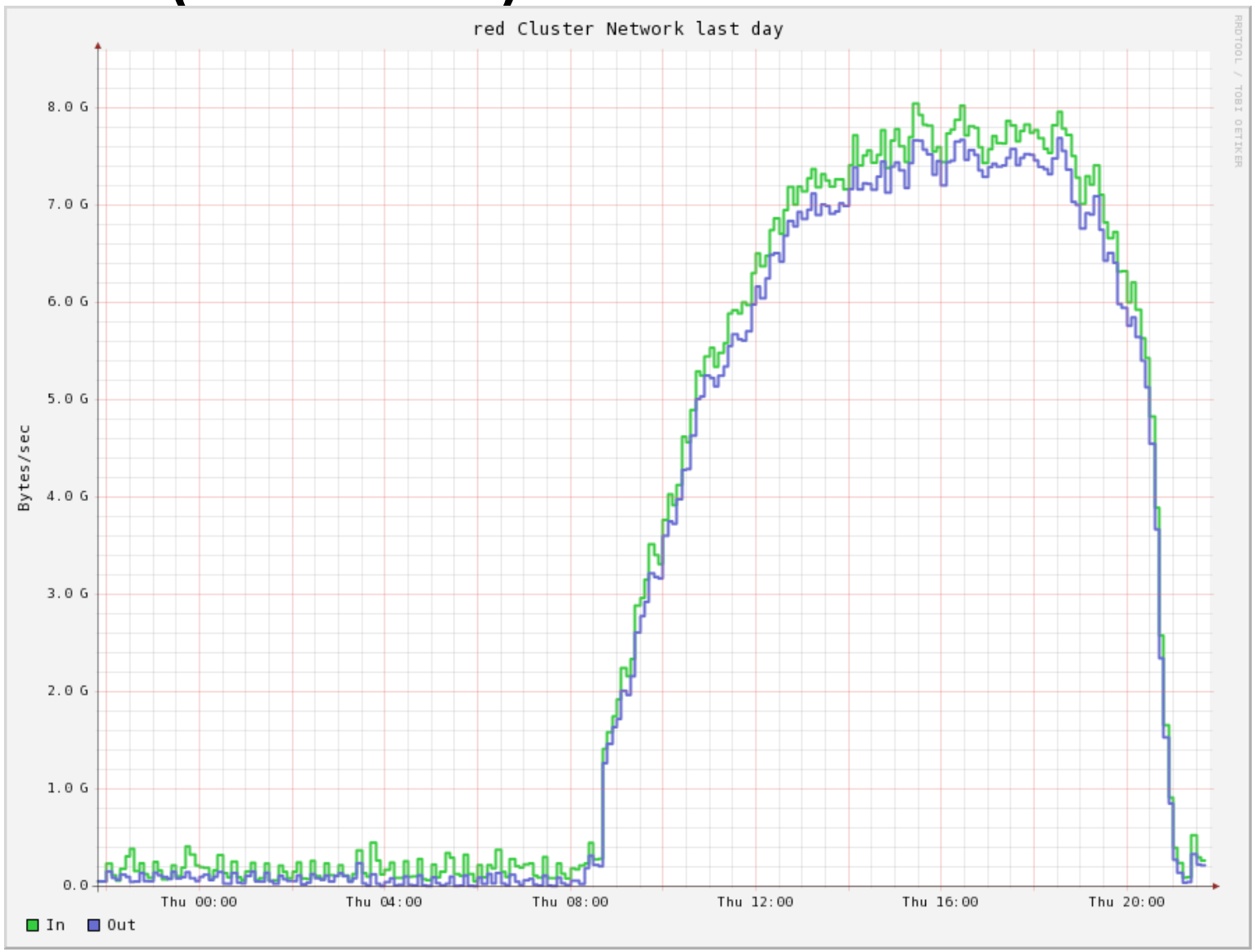
- Datanode deaths are detected when a heartbeat stops for X minutes; re-replication of missing data is wicked fast.



Usability

- Simple client tools for basic I/O operations (get, put, ls, mkdir, rm, mv, etc)
- Or just mount the filesystem read / write using FUSE and any POSIX client works*.
  - * seeks during write not supported
  - This is how CMS jobs run at our site. ROOT I/O is done directly against HDFS through POSIX.
- Users don't have to know *anything* about your filesystem.

(CMSSW) Performance



Performance

- We've clocked:
  - The filesystem at 80Gbps.
  - The SRM endpoints at 30Hz.
  - Fsck tool <10s.
  - Decommissioning in under an hour.
  - Namenode restart in about 60s.
  - WAN transfers peak at 9Gbps, sustain 5Gbps.

Future

- In production now! Our primary SE.
- Packaging and deployment.
- Perhaps putting together a hands-on session at UCSD this spring (?)
- A few “final touches” for code
  - Remove FUSE dependency for BeStMan server – use Java API directly.
  - There are other good research projects, but using it as a general-purpose SE is already production.