



Qserv

Prototype of the baseline LSST database architecture

Daniel L. Wang (SLAC), Serge M. Monkewitz (IPAC), Kian-Tat Lim (SLAC),
Jacek Becla (SLAC)



Requirements



- Incrementally scalable
- Reliable and available
- Low cost

Sustained concurrent load of
~50 interactive and 20 complex
queries

- Interactive: <10sec
- Object-based: <1h
- Source-based: <1d
- ForcedSource-based: <7d

Key table sizes (final data release)

Table	# rows	footprint
Object	26 billion	48TB
Source	2.1 trillion	1.3PB
ForcedSource	21 trillion	620TB



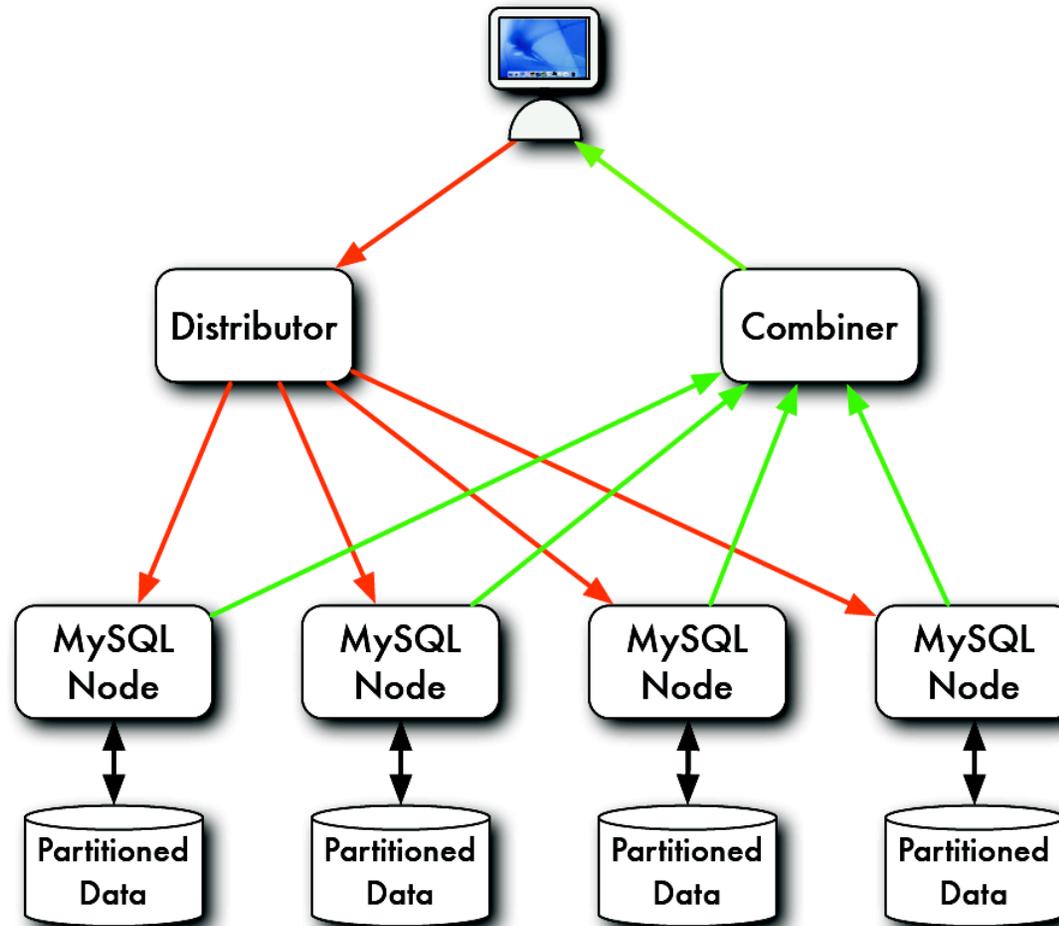
Baseline Architecture



- MPP RDBMS on shared-nothing commodity cluster
- Data clustered spatially and by-time, partitioned w/overlaps
 - Two-level partitioning
 - 2nd level materialized on-the-fly
 - Transparent to end-users
- Selective indices to speed up interactive queries, spatial searches, joins including time series analysis
- Shared scans
- Custom software based on open source RDBMS (MySQL) + xrootd

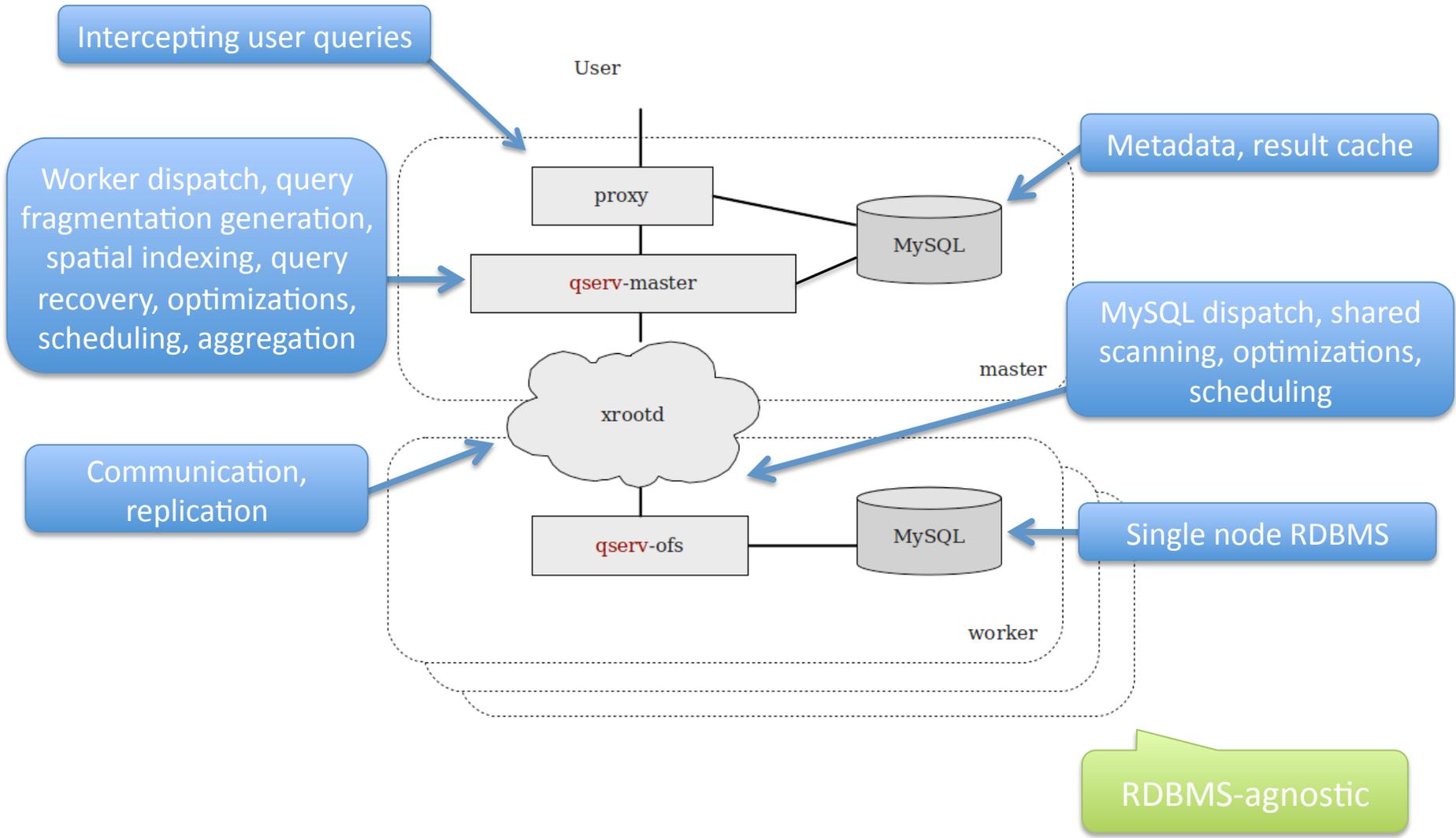


Baseline Architecture





Qserv Implementation

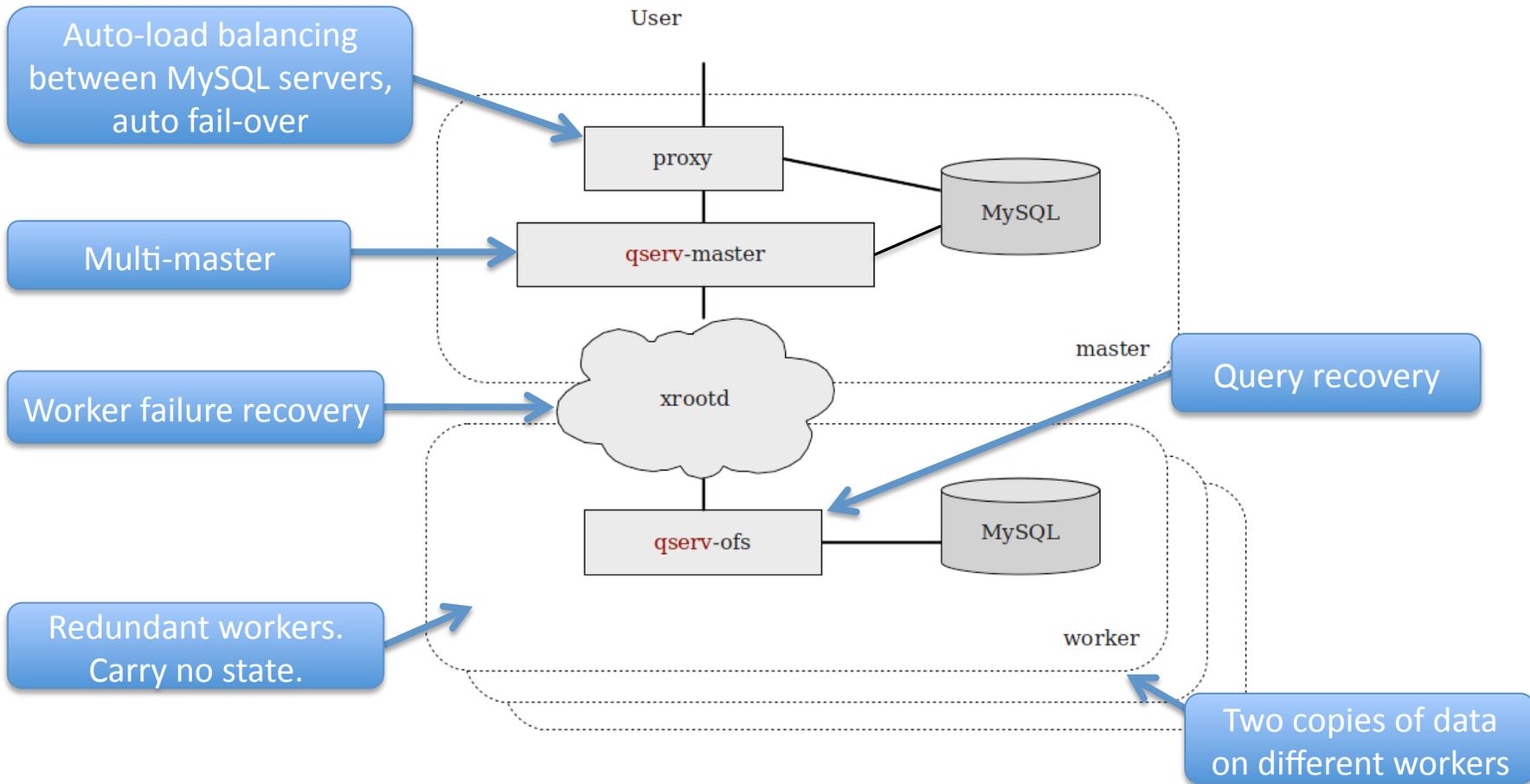




Qserv Fault Tolerance



- Components replicated
- Failures isolated
- Narrow interfaces
- Logic for handling errors
- Logic for recovering from errors





Partitioning/Spatial Queries



Spatial partitions

- Interactive small radius cone searches, near neighbor correlations $O(N^2) \rightarrow O(kN)$

Large number of partitions

- increases potential for parallelism
- simplifies rebalancing when adding/removing nodes.
- speeds up near neighbor correlations
- increases overhead

Two level partitioning on (α, δ) with overlap

- coarse partitions are unit of distribution across workers
- Fine partitions are materialized on the fly
- Partitions overlap so near neighbor queries avoid inter-worker communication
- Fine partitions provide (coarse) spatial index for small radius cone searches



Status



Scaling tests: 150 nodes, 2e9 objects, 55e9 sources (~32TB)

- ~4-9s: object retrieval, small area spatial search, object time series
- ~3-8m: full sky filter, densities (scans)
- ~10m – 5h: near neighbor, sources not near objects (joins)

Future Plans

- Shared scans
- User tables, cross-match
- Support for updates
- Query fault tolerance
- Partition management
- Usability improvements
- Authentication and authorization
- Resource management
- sciSQL: HTM and other UDFs?

Qserv docs: <http://dev.lsstcorp.org/trac/wiki/dbScalableArch>

Qserv code: <http://dev.lsstcorp.org/trac/browser/DMS/qserv>

sciSQL code: <https://launchpad.net/scisql>