SAND2008-5194C

Performing Fault-tolerant, Scalable Data Collection and Analysis

James Jolly, Intern
University of Wisconsin-Madison

David Thompson, Mentor
Distributed Visualization Systems
Sandia National Laboratories / CA

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under contract DE-AC04-94AL85000.

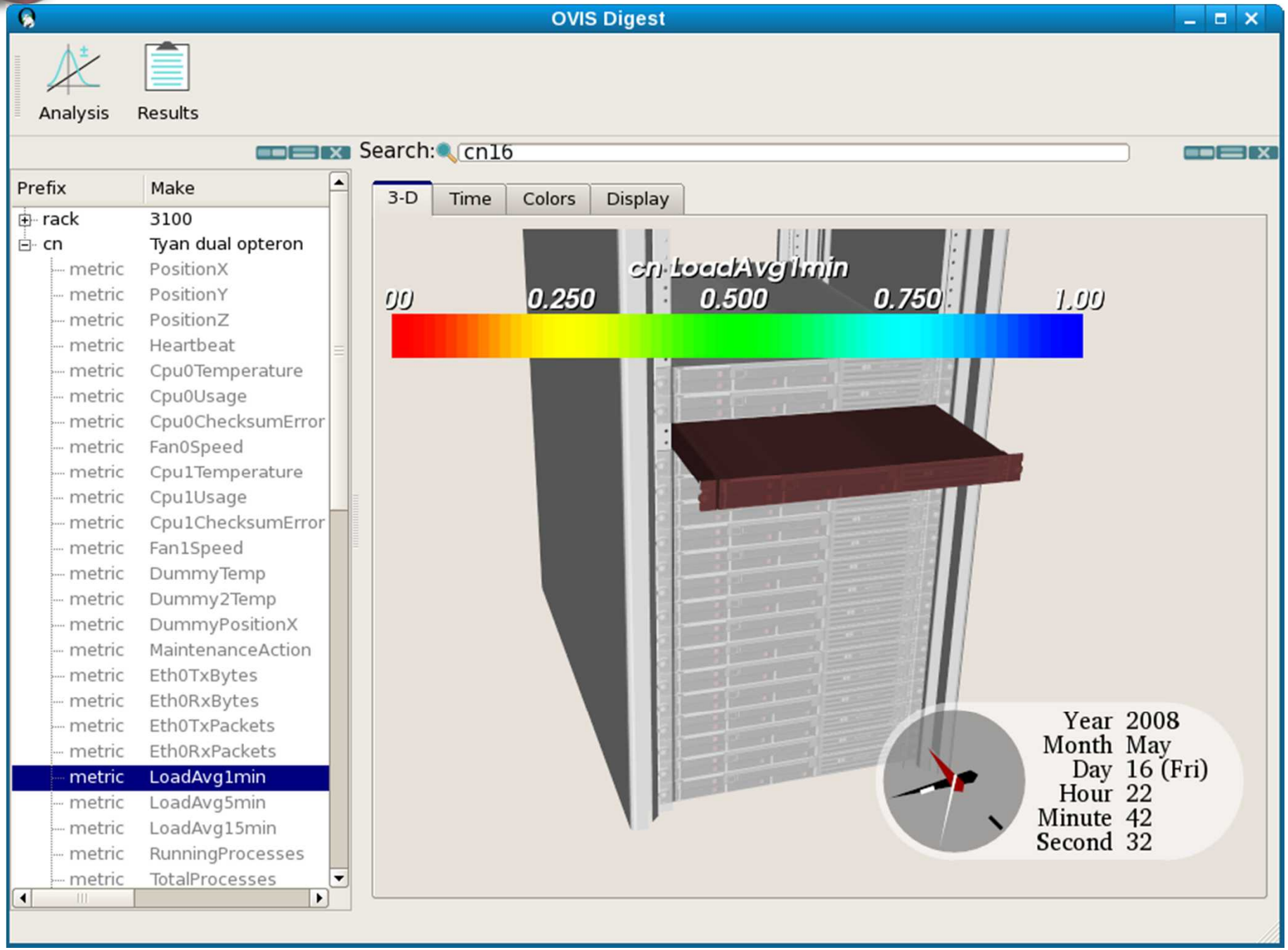
Problem Domain

- need to continuously collect data from many similar devices
  examples:
  - computational clusters
  - sensor networks
- instrumentation does not interfere with operation
- continuously mine stored data for models, patterns, or trends
- results are continuously recorded
- users can visualize data and results as they are generated

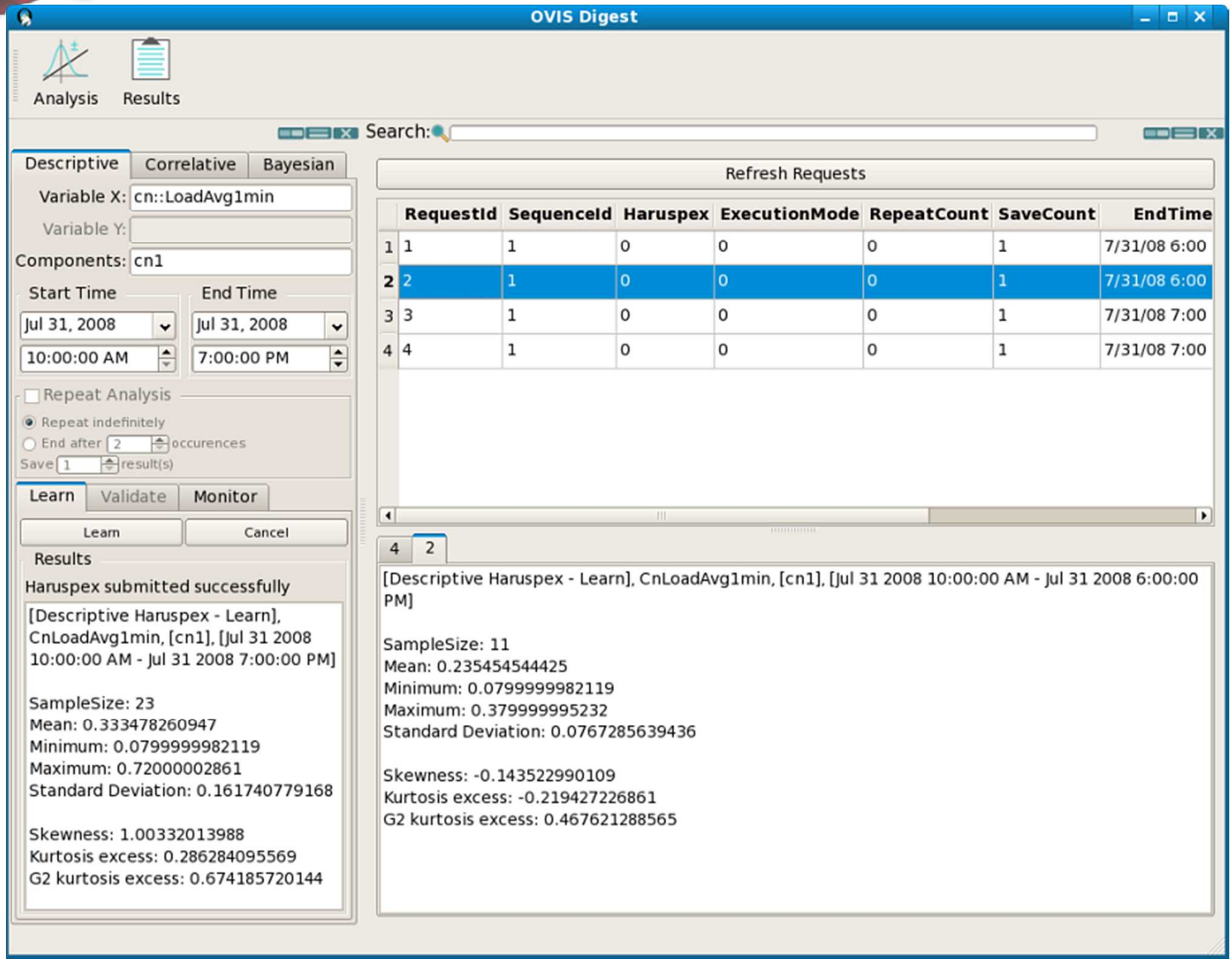
The OVIS Approach

- users define instrumentation
  - unique identifiers for devices
  - what metrics to log for these devices
  - their physical layout
- multi-tier application
- collection agents (sheep) record metric data in relational database
- analysis engines (shepherds) perform analyses in parallel and asynchronous fashion
- visualization (baron) schedules statistical analyses and depicts results within the context of the layout

The OVIS Approach



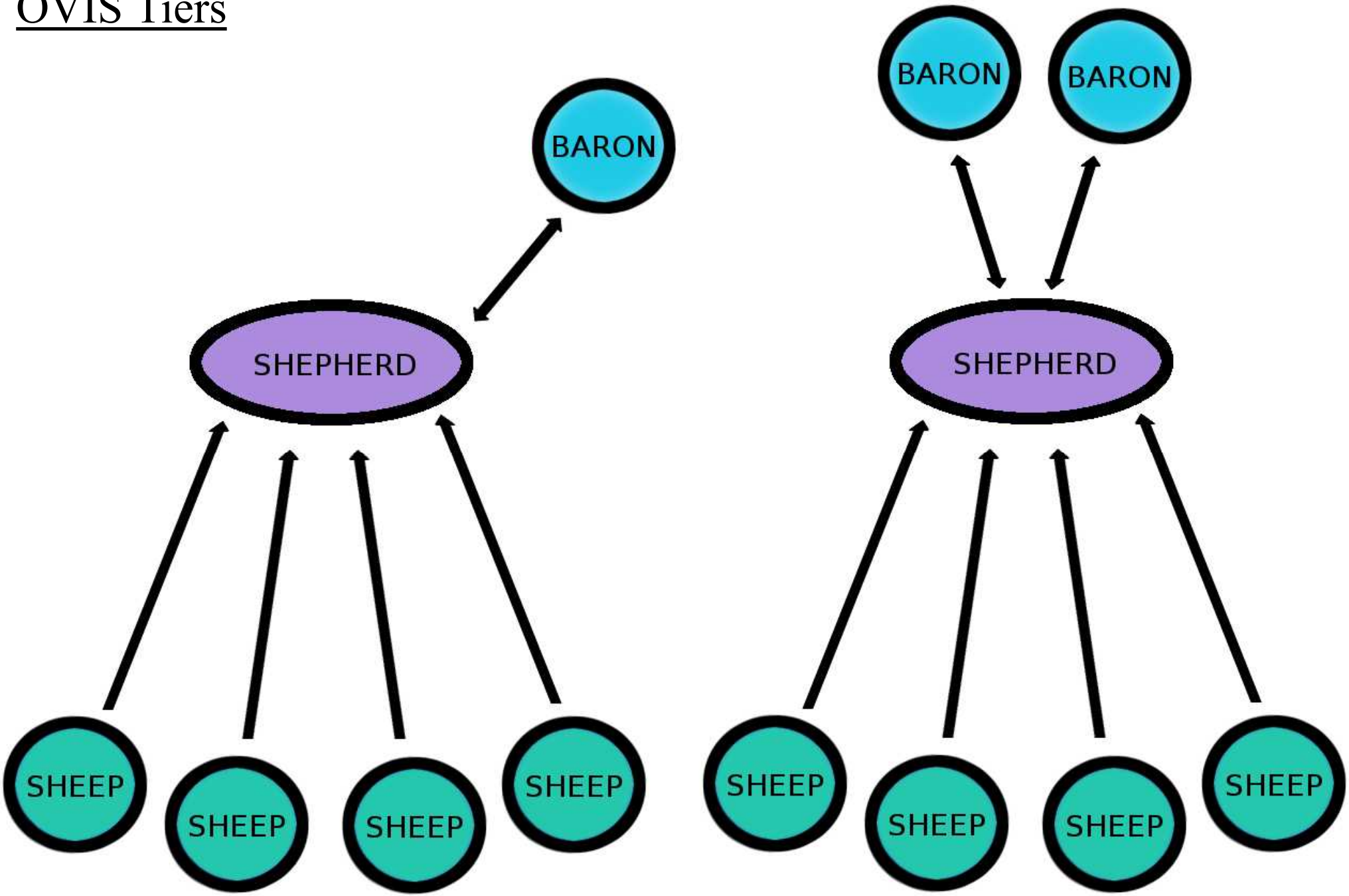
The OVIS Approach



Magic Behind the Scenes

- application logic vertically partitioned
- agents publish and subscribe to services themselves
- shared-nothing system

OVIS Tiers


BARON
SHEPHERD
SHEEP
SHEEP
SHEEP
SHEEP
BARON
BARON
SHEPHERD
SHEEP
SHEEP
SHEEP
SHEEP

How do we organize the data?

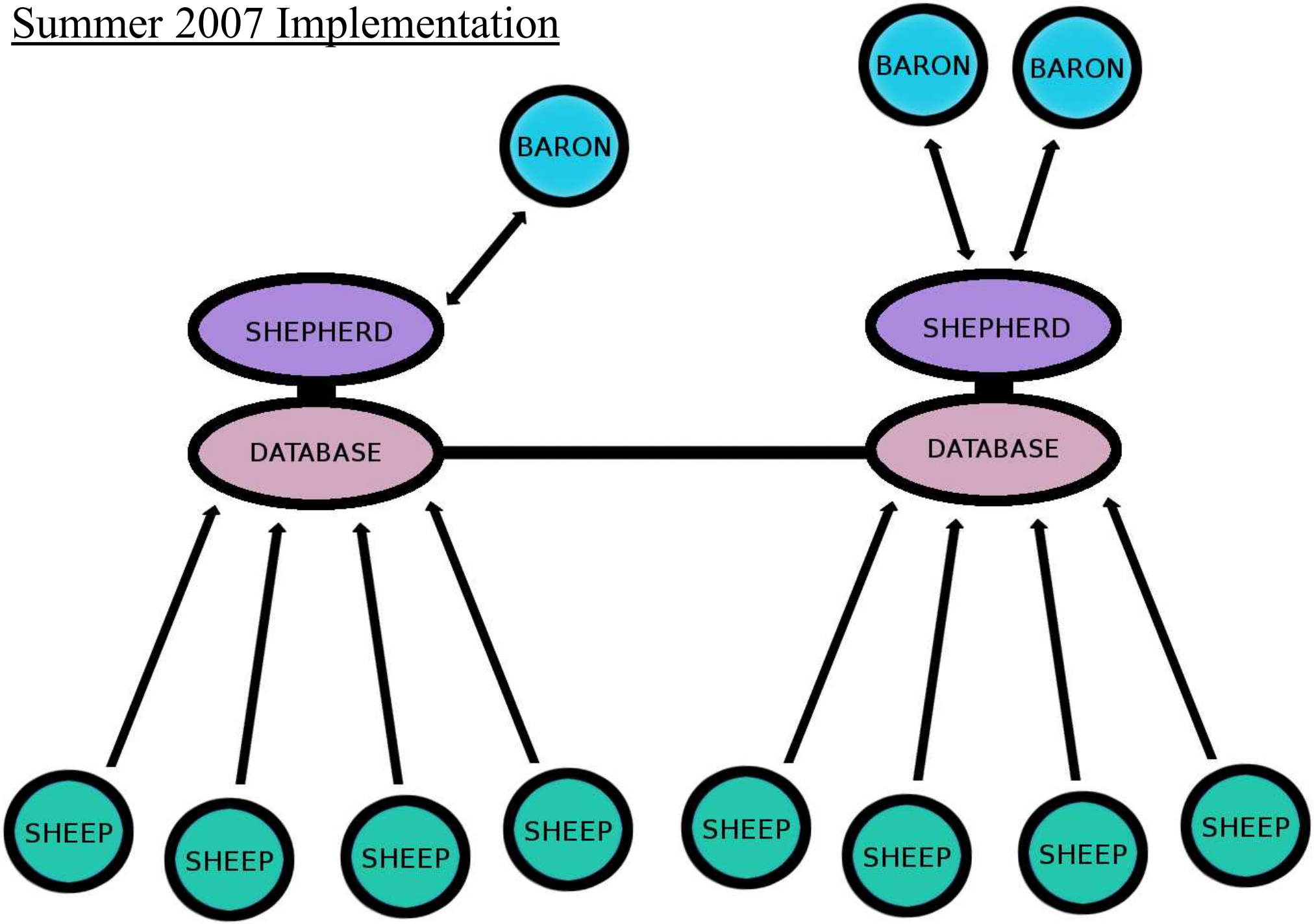
- shepherds must process different data in parallel, merge results
- DBMS can not reside on sheep

  ... place databases on machines running shepherd

- barons should see same results connecting to any shepherd

  ... replicate databases

Summer 2007 Implementation



Problems With Naïve Replication

- not enough disk space to replicate everything
- message-passing overhead

... naïve database replication will not scale

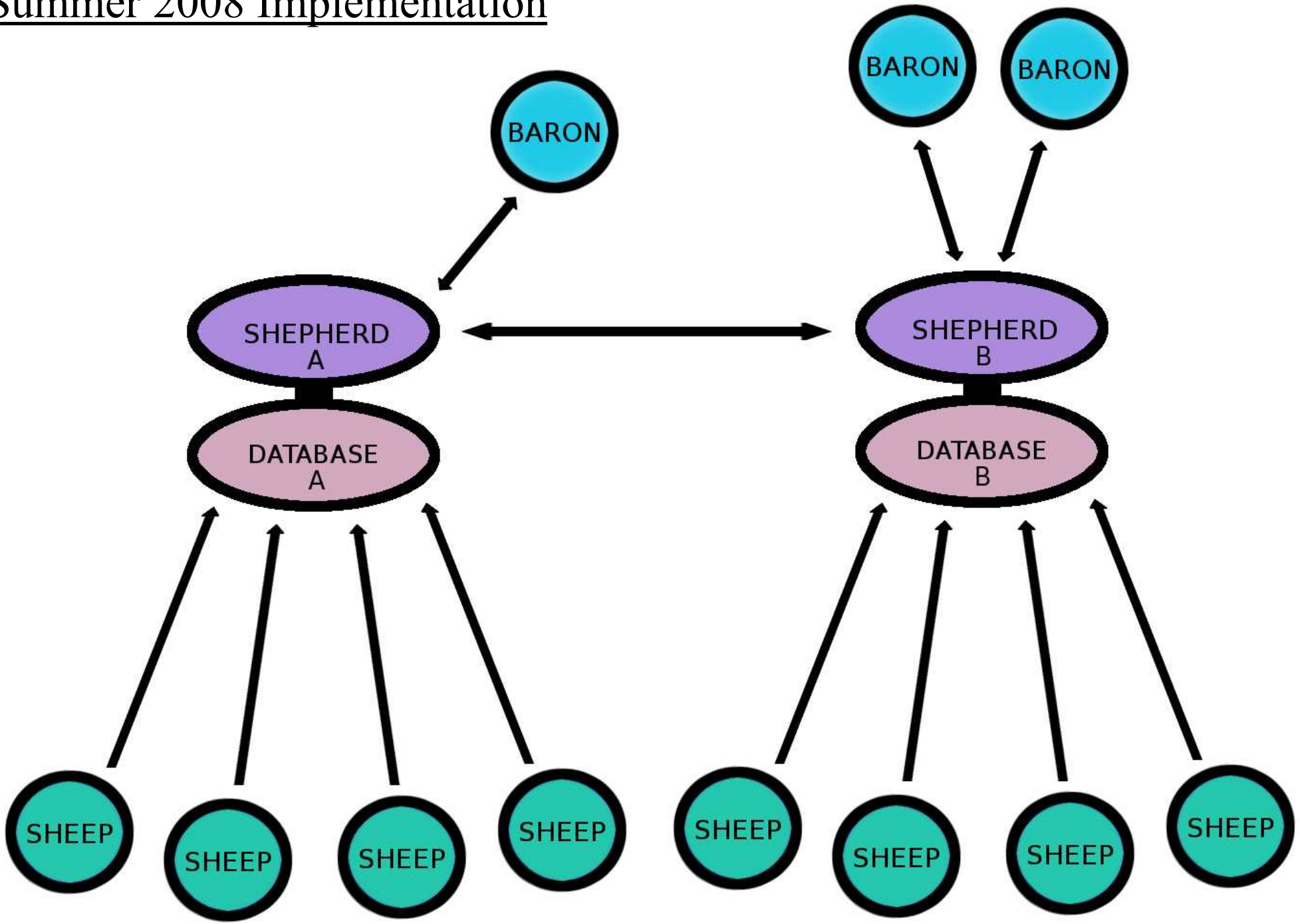
... horizontal table partitioning will scale

MySQL Cluster Supports Partitioning

but suffers from...

- high communication overhead
  - synchronous replication with two-phase commit
- inability to expand dynamically
- table contents remaining in memory
- licensing issues

Summer 2008 Implementation

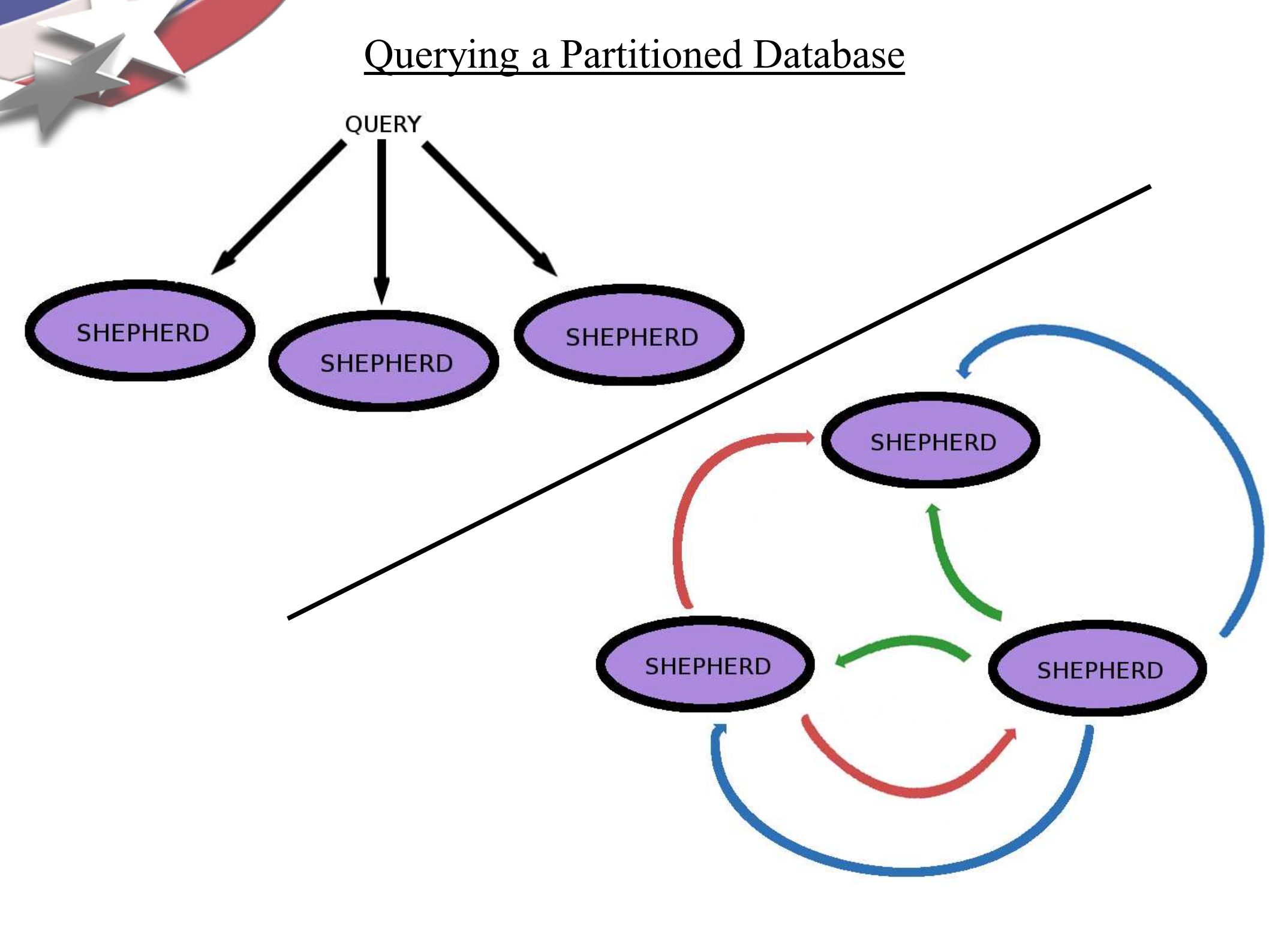


Querying a Partitioned Database

each shepherd ...

- stores data for only the sheep that connect to it
  - note metric data for a device may be split across many shepherds
- upon query, calculates statistical moments from this data
- shares local moments with all other shepherds
- merges moments from all other shepherds with local moments

Querying a Partitioned Database



Advantages of New Implementation

- lightweight replication
- asynchronous
- shepherd failures still yield partial results

Possible Future Work

- quantify certainty in partial results
- parallelize visualization for use with multiple database backend
- efficiently aggregate data using a tree-based overlay network
- use hierarchical dataset sub-sampling

Thanks!