My other computer is a datacentre

Steve Loughran

Julio Guijarro

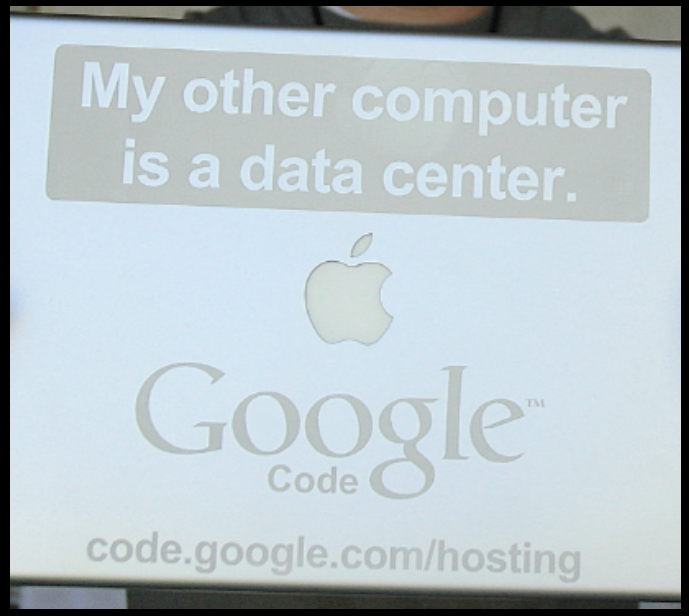
December 2010

Our other computer is

a datacentre

His other computer is a datacentre

- Open source projects
- SVN, web, defect tracking
- Released artifacts



code.google.com

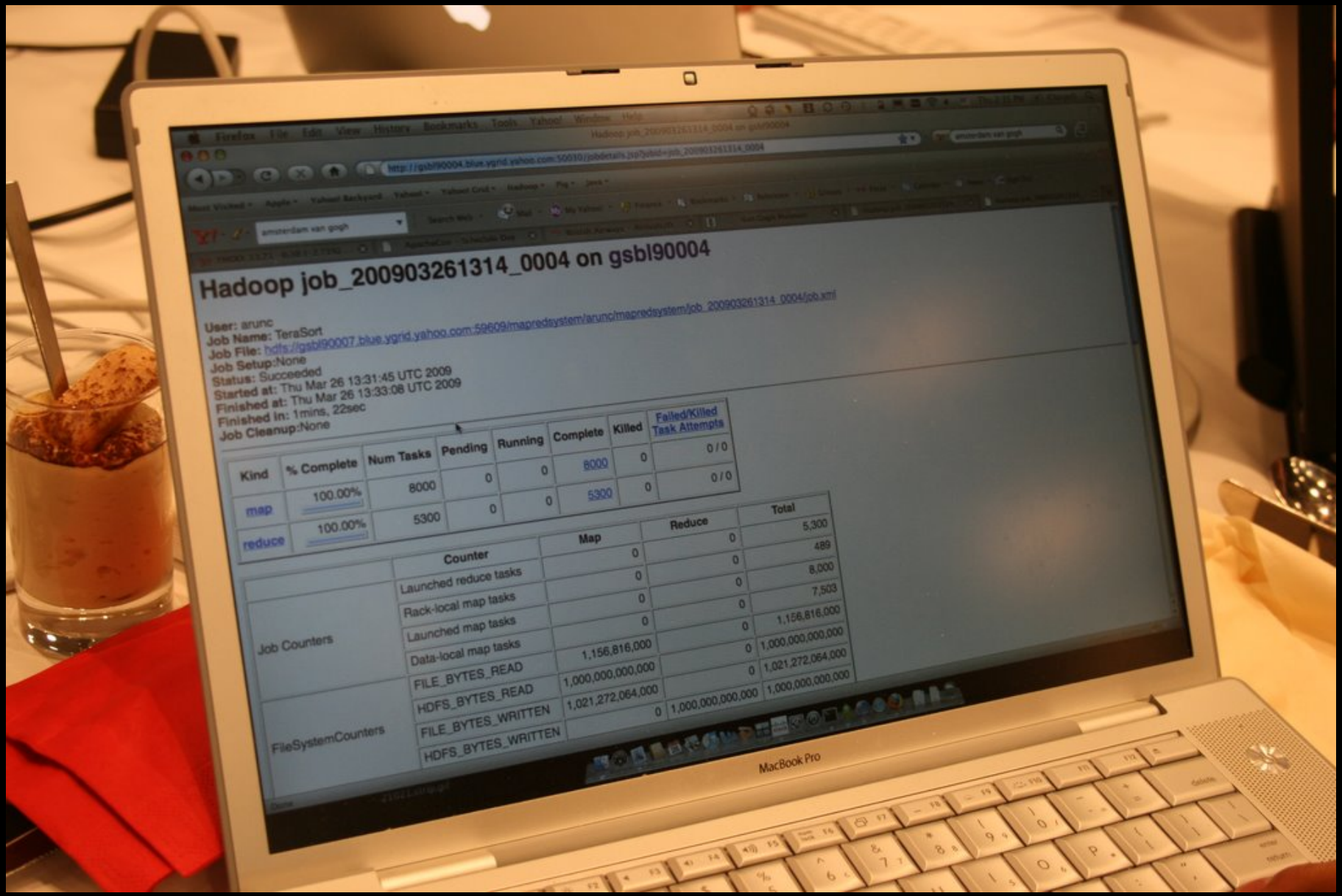
His other computer is a datacentre

- Email
- Videos on YouTube
- Flash-based games

Their other computer is a datacentre

Owen and Arun at Yahoo!

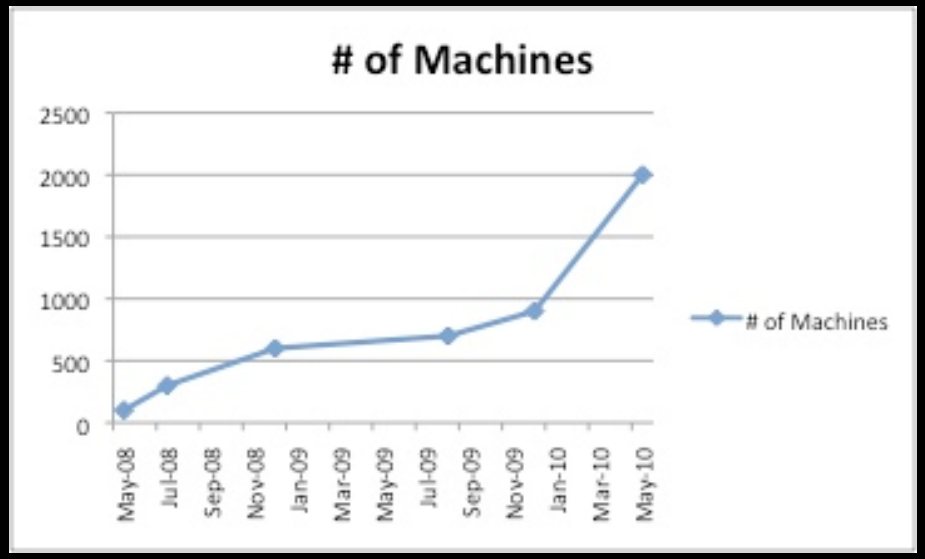
That sorts a Petabyte in 82s

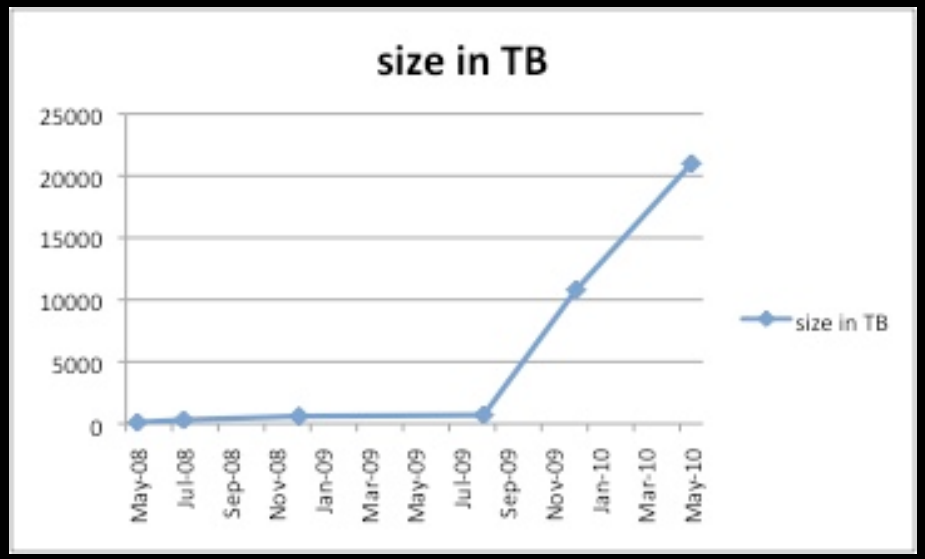


This datacentre

Yahoo! 8000 nodes, 32K cores, 16 Petabytes

His other computer is a datacentre





Dhruba at Facebook!

Problem: Big Data

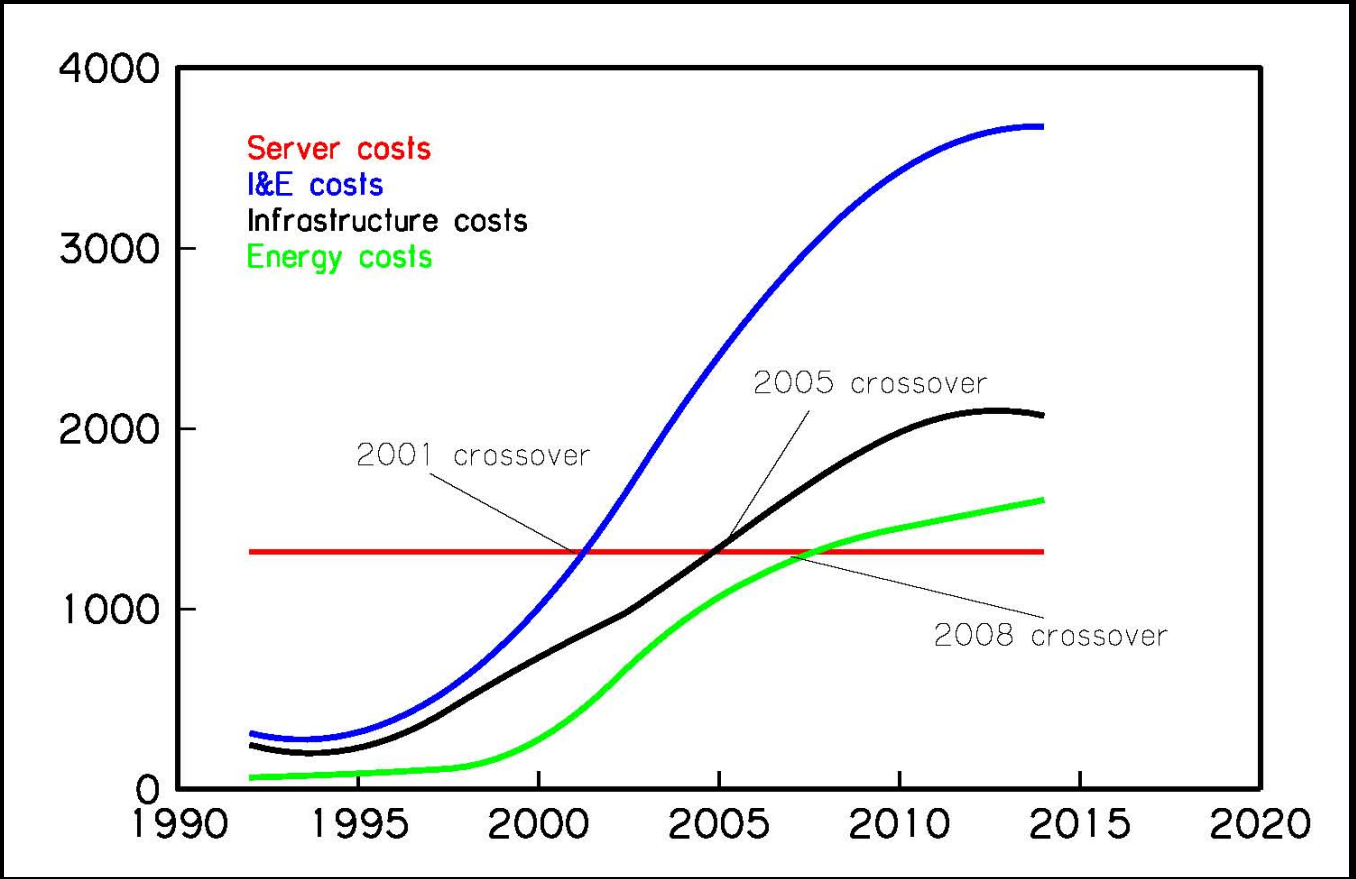
Storing and processing PB of data

Cost-effective storage of Petabytes

- Shared storage+compute servers
- Commodity hardware (x86, SATA)
- Gigabit networking to each server
- Redundancy in the application, not RAID

Move problems of failure to the app, not H/W

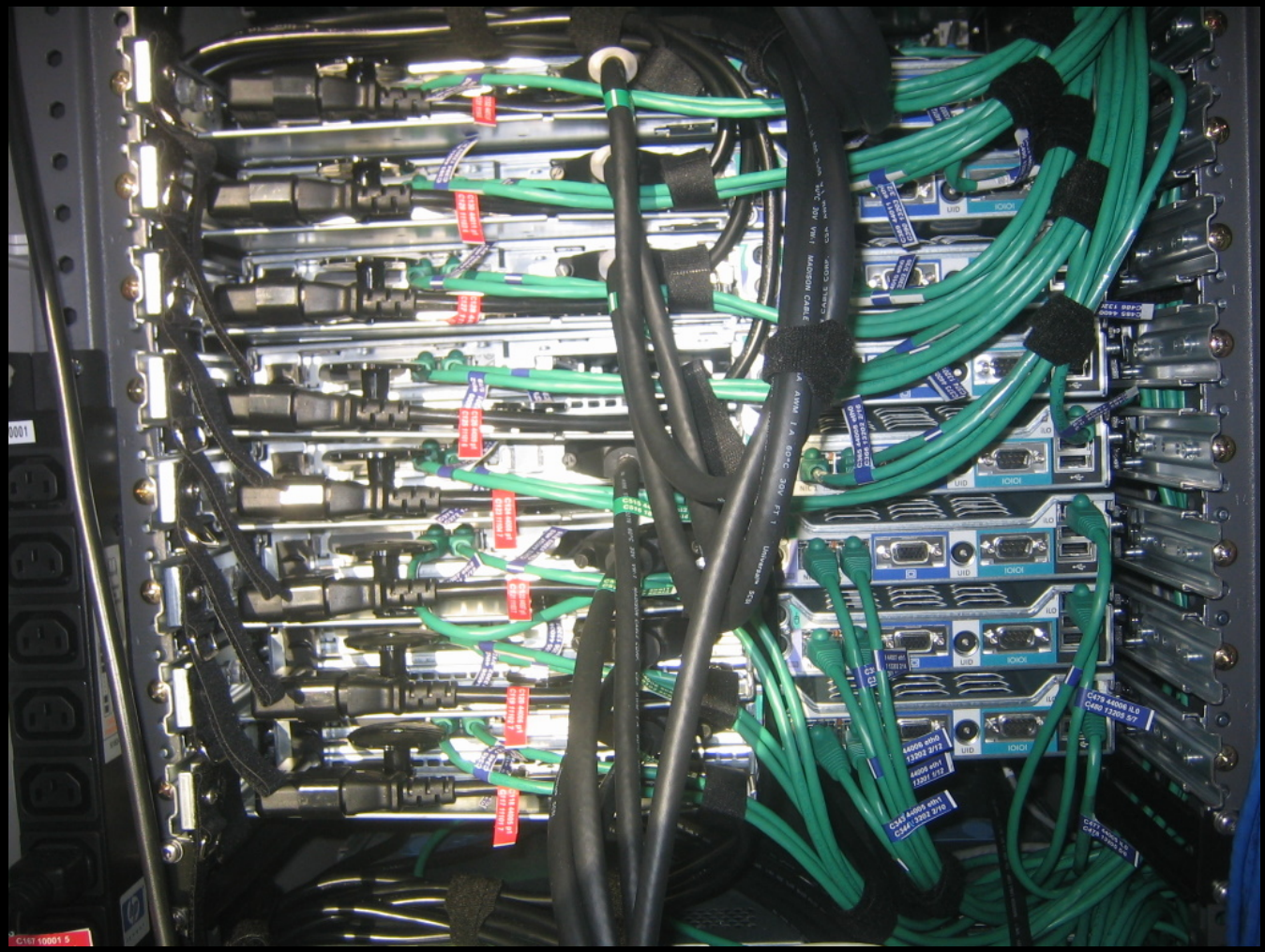
There are no free electrons



Power

- HVAC power 1.5-2X system power
  =server W saved has follow-on benefits
- Idle servers still consume 80% peak power
  =keep busy or shut down
- Gigabit copper networking costs power
- SSD front end machines save on disk costs, and can be started/stopped fast.

Hardware



Network Fabric

- 1Gbit/s from the server
- 10 Gbit/s between racks
- Two 10Gb/s for failover
- Multiple off-site links

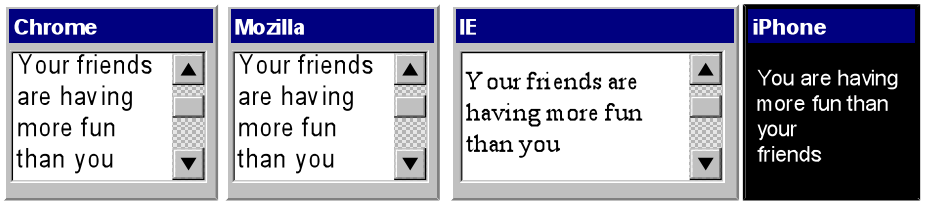
Bandwidth between racks is a bottleneck

Where?

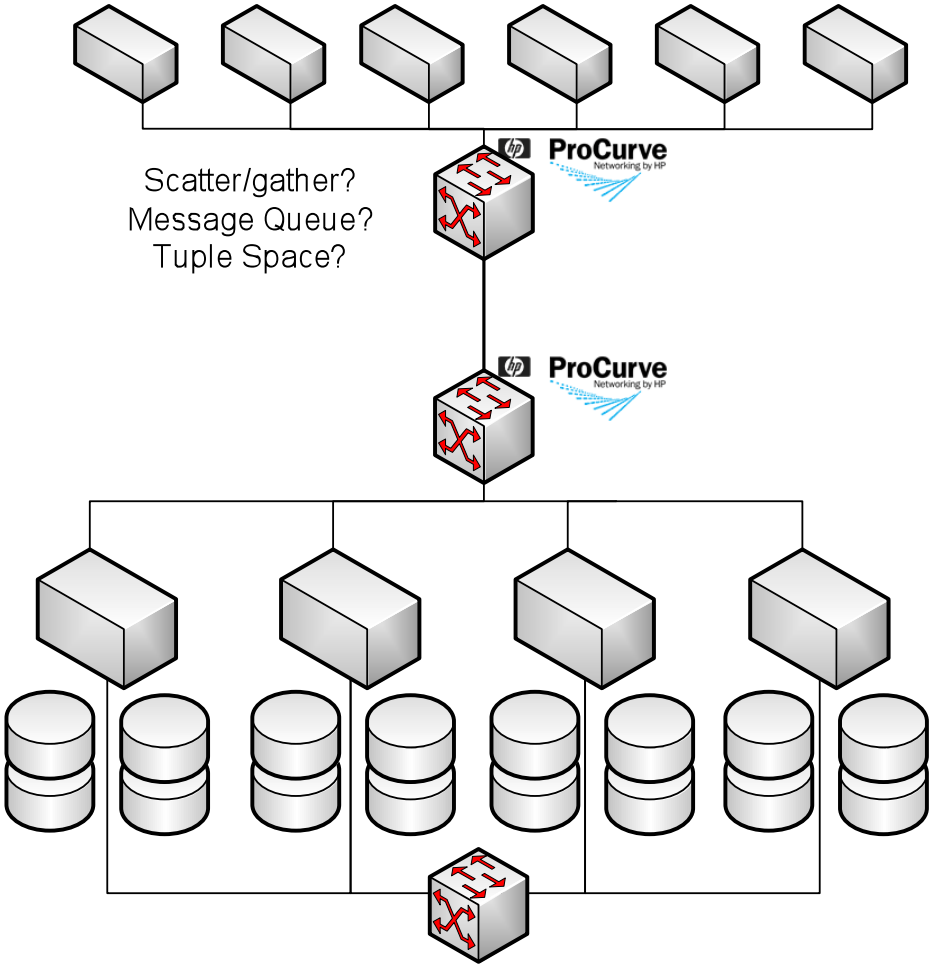
- Low cost (Hydro) electricity
- Cool and dry outside air (for open air cooling)
- Space for buildings
- Networking: fast, affordable, >1 supplier
- Low risk of earthquakes and other disasters
- Hardware: easy to get machines in fast
- Politics: tax, govt. stability/trust; data protection

Oregon and Washington States

Trend: containerized clusters



REST APIs



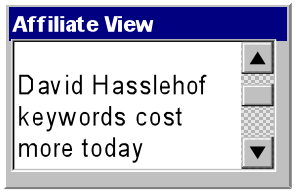
Scatter/gather?
Message Queue?
Tuple Space?

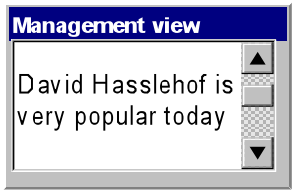
Diskless front end

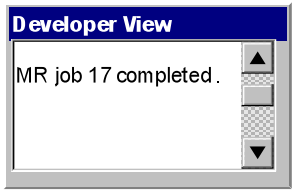
Memcached
JSP?
PHP?

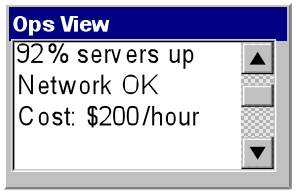
Cloud Layer

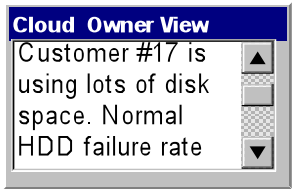
HDFS
HBase
MapReduce
DLucene











High Availability

- Avoid SPOFs
- Replicate data.
- Redeploy applications onto live machines
- Route traffic to live front ends
- Decoupled connection to back end: queues, scatter/gather
- Issue: how do you define *live*?

Web Front End

- Disposable machines
- Hardware: SSD with low-power CPUs
- PHP, Ruby, JavaScript: agile languages
- HTML, Ajax
- Talk to the back end over datacentre protocols

Work engine

- Move work to where the data is
- Queue, scatter/gather or tuple-space
- Create workers based on demand
- Spare time: check HDD health or power off
- Cloud Algorithms: MapReduce, BSP

MapReduce: Hadoop

Highbury Vaults Bluetooth Dataset

```
{lost,"00:0F:B3:92:05:D3","2008-04-17T22:11:15",1124313075}
{found,"00:0F:B3:92:05:D3","2008-04-17T22:11:29",1124313089}
{lost,"00:0F:B3:92:05:D3","2008-04-17T22:24:45",1124313885}
{found,"00:0F:B3:92:05:D3","2008-04-17T22:25:00",1124313900}
{found,"00:60:57:70:25:0F","2008-04-17T22:29:00",1124314140}
```

- Six months worth of discovered Bluetooth devices
- MAC Address and timestamp
- One site: a few megabytes
- Bath Experiment: multiple sites

MapReduce to Day of Week

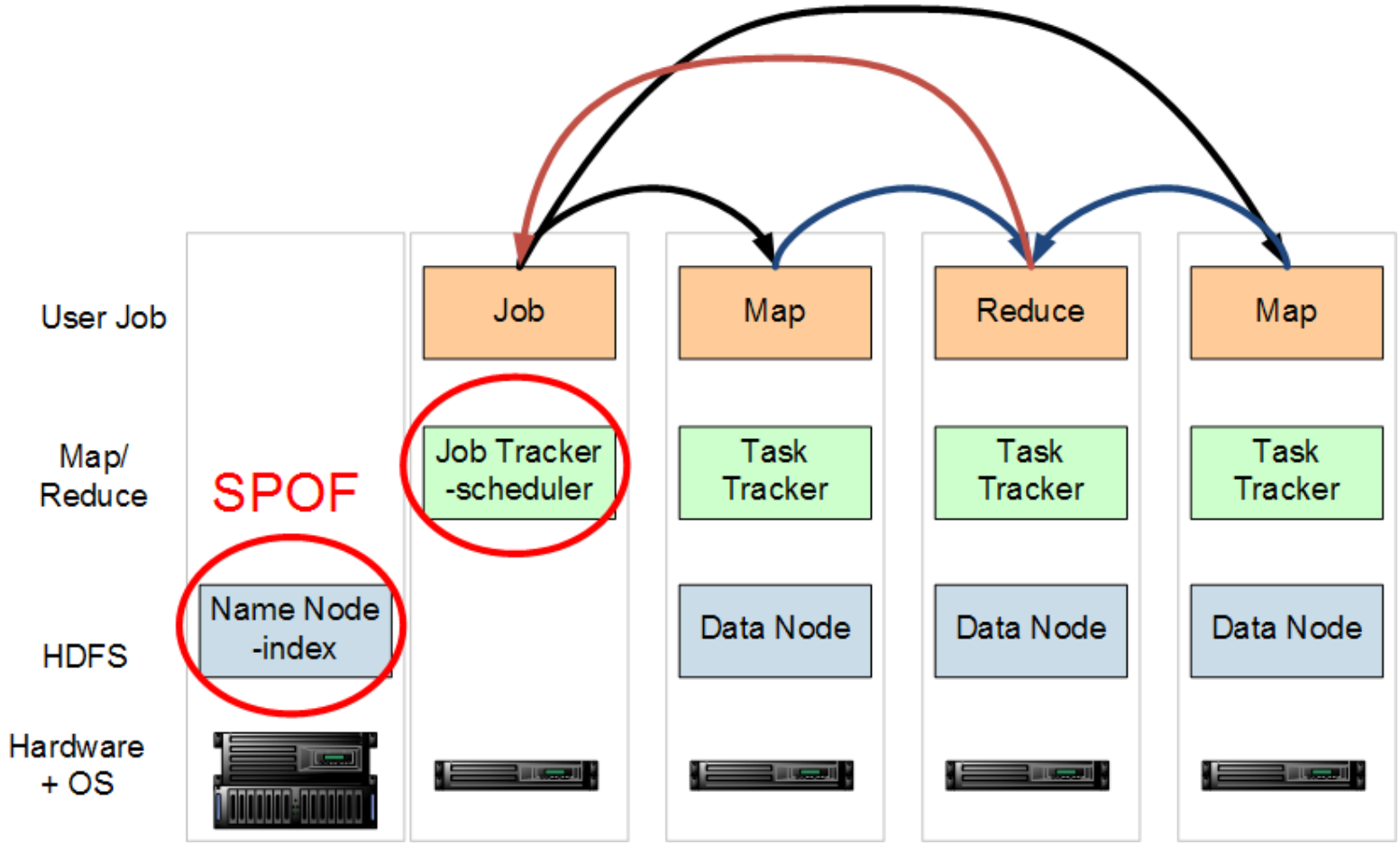
```
map_found_event_by_day_of_week(
 {event, found, Device, _, Timestamp}, Reducer) ->
  DayOfWeek = timestamp_to_day(Timestamp),
  Reducer ! {DayOfWeek, Device}.

size(Key, Entries, A) ->
  L = length(Entries),
  [ {Key, L} | A].

mr_day_of_week(Source) ->
  mr(Source,
    fun traffic:map_found_event_by_day_of_week/2,
    fun traffic:size/3,
    []).
```

Results

```
traffic:mr_day_of_week(big).
[{3,3702},
 {6,3076},
 {2,3747},
 {5,3845},
 {1,3044},
 {4,3850},
 {7,2274}]
```

Monday	3044
Tuesday	3747
Wednesday	3702
Thursday	3850
Friday	3845
Saturday	3076
Sunday	2274

Hadoop running MapReduce



Filesystem

- Commodity disks
- Scatter data across machines
- Duplicate data across machines
- Trickle-feed backup to other sites
- Long-haul API: HTTP
- Archive unused files
- In idle time: check health of data, rebalance
- *Monitor and predict disk failure.*

Logging & Health

- Everything will fail
- There is always an SPOF
- Feed the system logs into the data mining infrastructure: post-mortem and prediction

Monitor and mine the infrastructure
-with the infrastructure

What will your datacentre do?

Management

- JMX for Java Code
- Nagios
- Jabber: use google talk and similar to monitor.
- Future: X-trace, Chukwa
- Feed the logs into the filesystem for data mining

Monitor and mine infrastructure

Infrastructure Layer

- Public API: create VM images, deploy them
- Internal: billing for CPU, network use
- Monitoring: who is killing our network?
- Prediction: what patterns of use will tie into purchase plans

Data mining of all management data

Testing

- Bring up a test cluster during developer working hours
- Run local tests (Selenium, HtmlUnit, JMeter) in datacentre to eliminate latency issues
- Run remote tests in a different datacentre to identify latency problems
- Push results into the filesystem *and mine*