University of Virginia

CS4740
CLOUD COMPUTING

Cloud Infrastructure in Industry

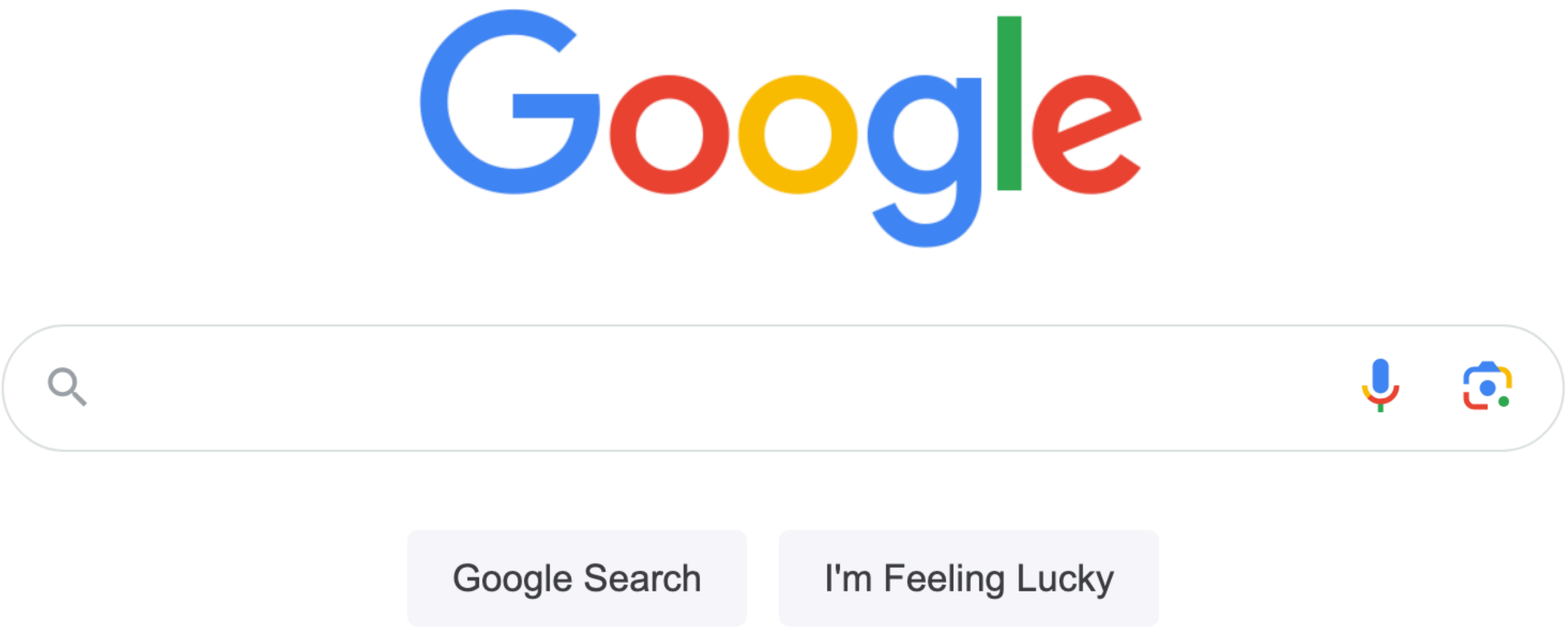
Prof. Chang Lou, UVA CS, Spring 2024

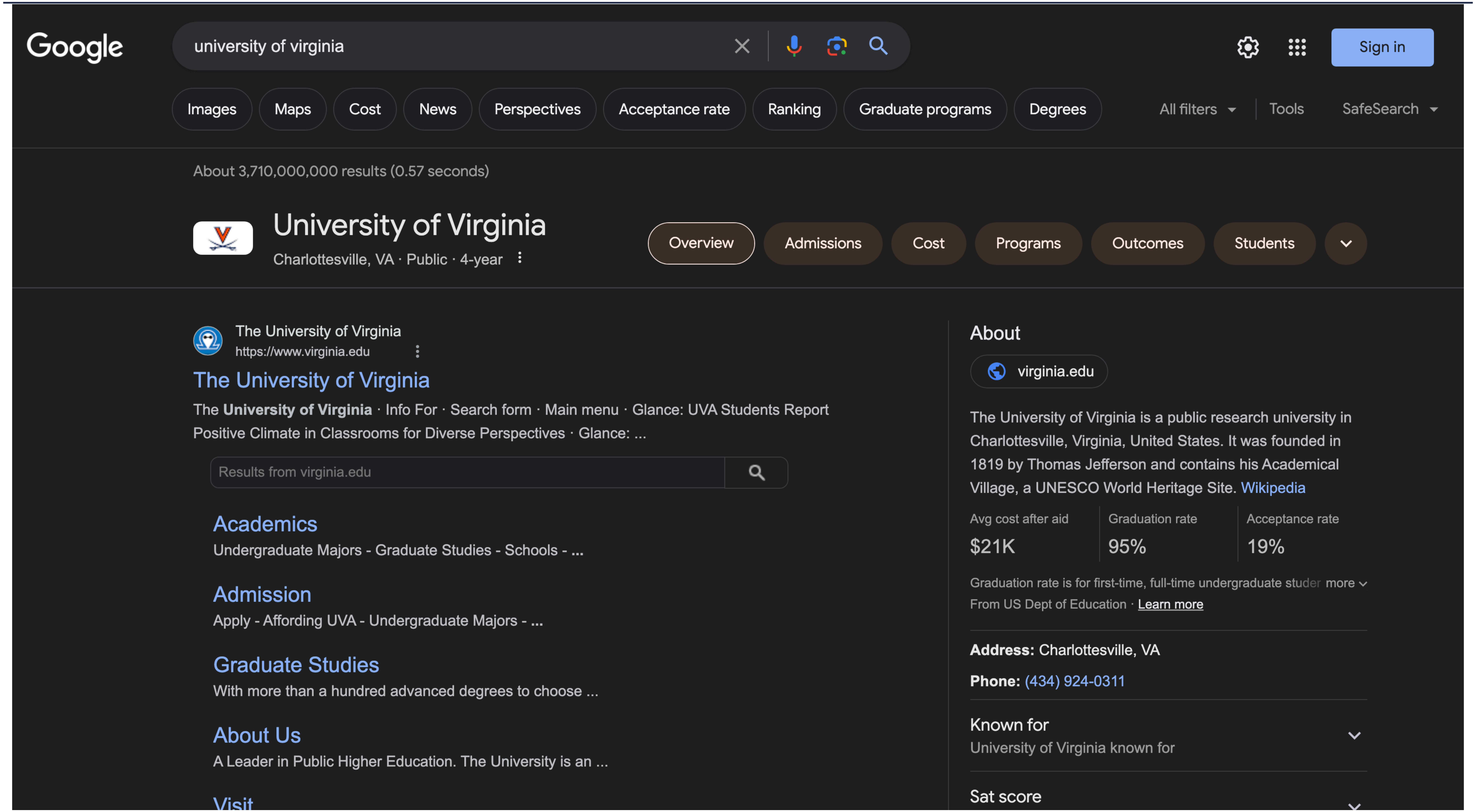

SIGAI at UVA
a part of ACM at UVA
acm
ARTIFICIAL INTELLIGENCE
TOWN HALL
AN INTERDISCIPLINARY DISCUSSION
AI and You: Navigating the Future Together at UVA and in the Broader World
EXPERT PANELISTS
PROF. MADHUR BEHL
Computer Science
PROF. REZA MOUSAVI
Commerce
PROF. MICHAEL PALMER
Center for Teaching Excellence, Director
TUESDAY
16th April, 2024
5:30 - 7:00 PM
NEWCOMB THEATRE
Located in Lower Level
Below Pavilion XI
REGISTER NOW
FREE SNACKS
Questions/Comments
sigai-contact@virginia.edu
sigai-at-uva.org
NEWCOMB HALL

LECTURE THEME

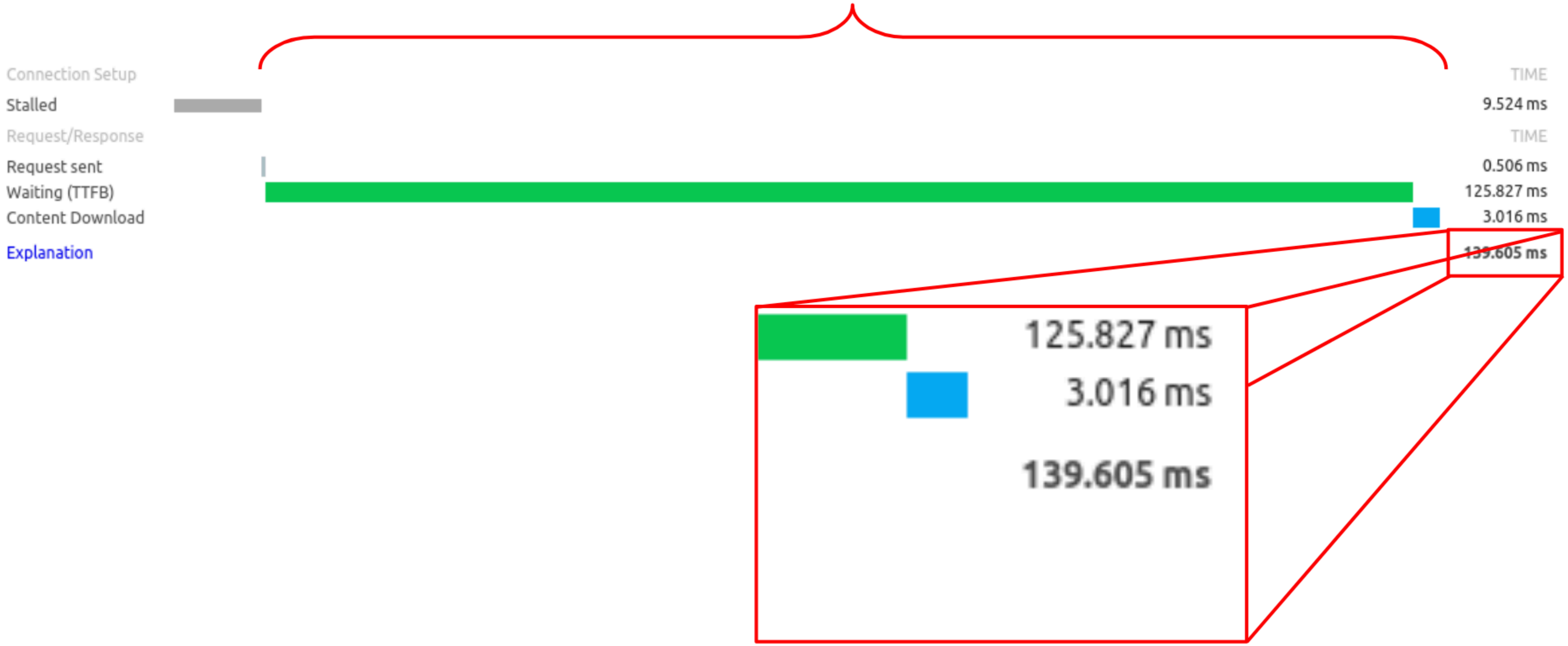
- We talked a lot about storage in this class, plus a bit about distributed computation. For storage, we focused on a particular type of interface (transactional databases).
- But there's a vast range of infrastructural components that are needed for building successful distributed applications. Large companies and open-source communities have such components available.
- This lecture aims to provide an index of such components. We won't give details about how these components are built, but pointers to where you can find out more.
  - the contents are heavily based on Malte Schwarzkopf's talk at Cambridge

“WHAT IT TAKES TO BUILD GOOGLE?”

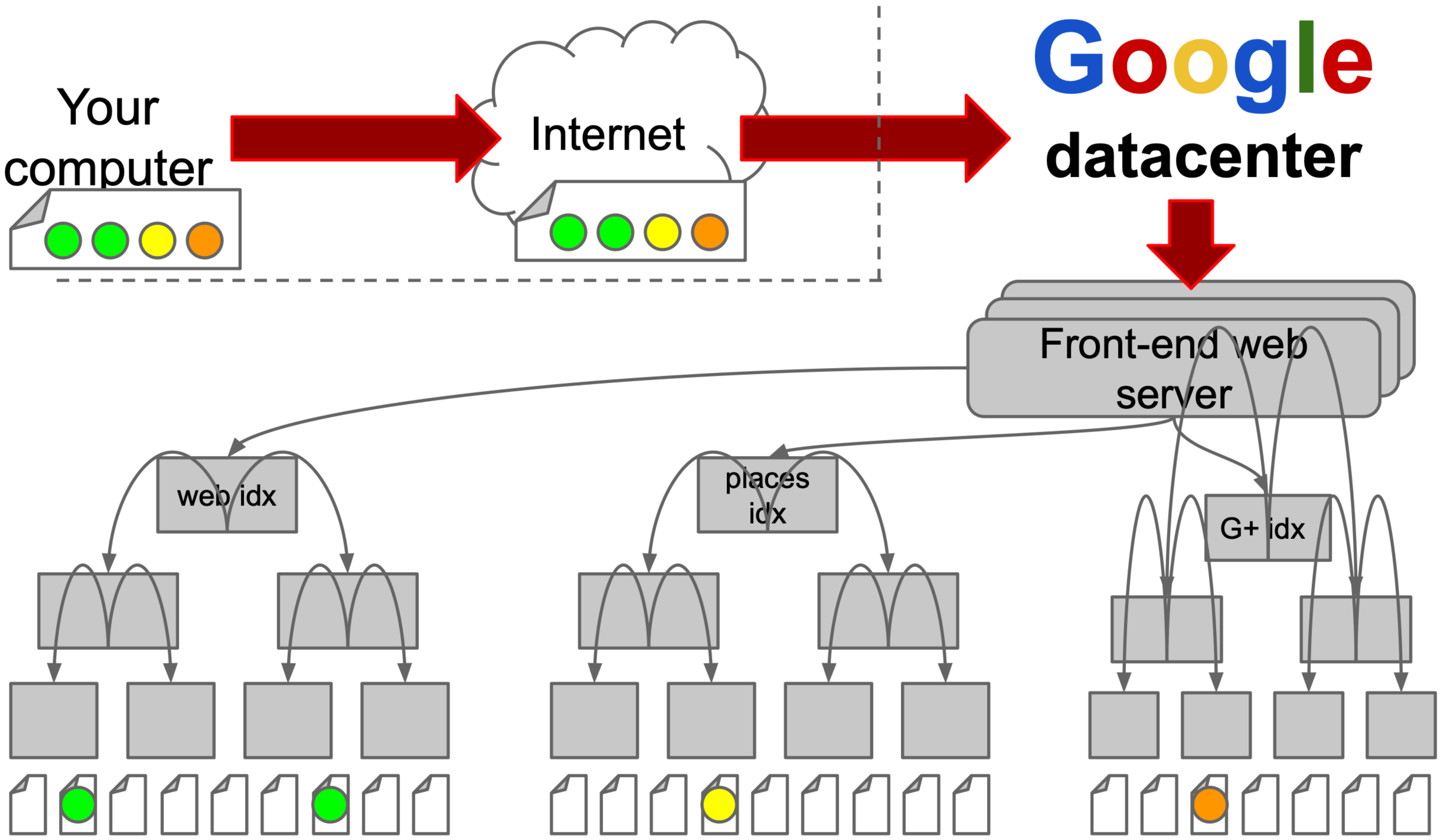



Google
university of virginia
Sign in
Images
Maps
Cost
News
Perspectives
Acceptance rate
Ranking
Graduate programs
Degrees
All filters
Tools
SafeSearch
About 3,710,000,000 results (0.57 seconds)
University of Virginia
Charlottesville, VA · Public · 4-year
Overview
Admissions
Cost
Programs
Outcomes
Students
The University of Virginia
https://www.virginia.edu
The University of Virginia
The University of Virginia · Info For · Search form · Main menu · Glance: UVA Students Report Positive Climate in Classrooms for Diverse Perspectives · Glance: ...
Results from virginia.edu
Academics
Undergraduate Majors - Graduate Studies - Schools - ...
Admission
Apply - Affording UVA - Undergraduate Majors - ...
Graduate Studies
With more than a hundred advanced degrees to choose ...
About Us
A Leader in Public Higher Education. The University is an ...
Visit
About
virginia.edu
The University of Virginia is a public research university in Charlottesville, Virginia, United States. It was founded in 1819 by Thomas Jefferson and contains his Academical Village, a UNESCO World Heritage Site. Wikipedia
Avg cost after aid
$21K
Graduation rate
95%
Acceptance rate
19%
Graduation rate is for first-time, full-time undergraduate studer more
From US Dept of Education · Learn more
Address: Charlottesville, VA
Phone: (434) 924-0311
Known for
University of Virginia known for
Sat score

What happens here?



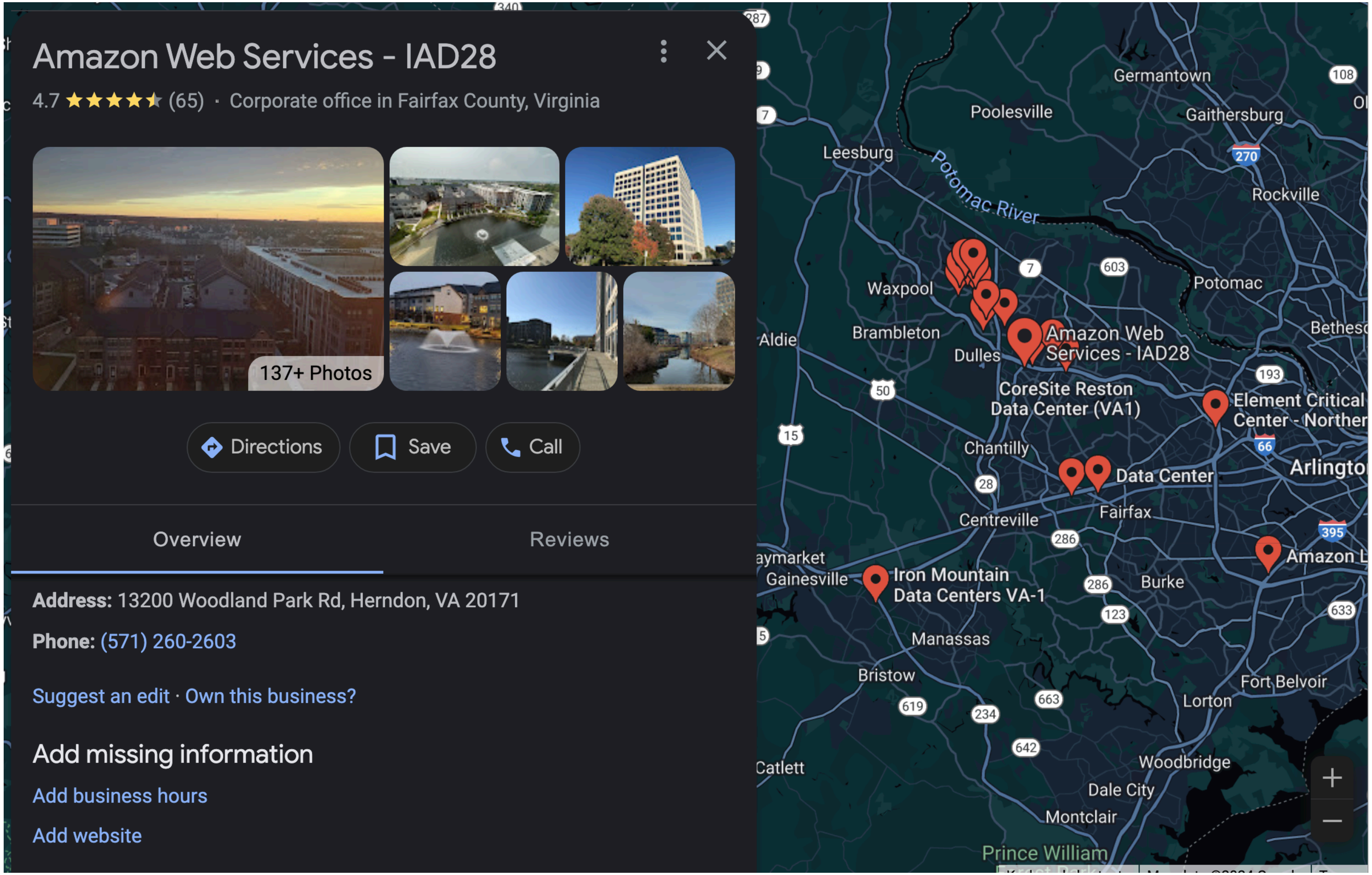
WHAT HAPPENS IN THOSE 139MS?

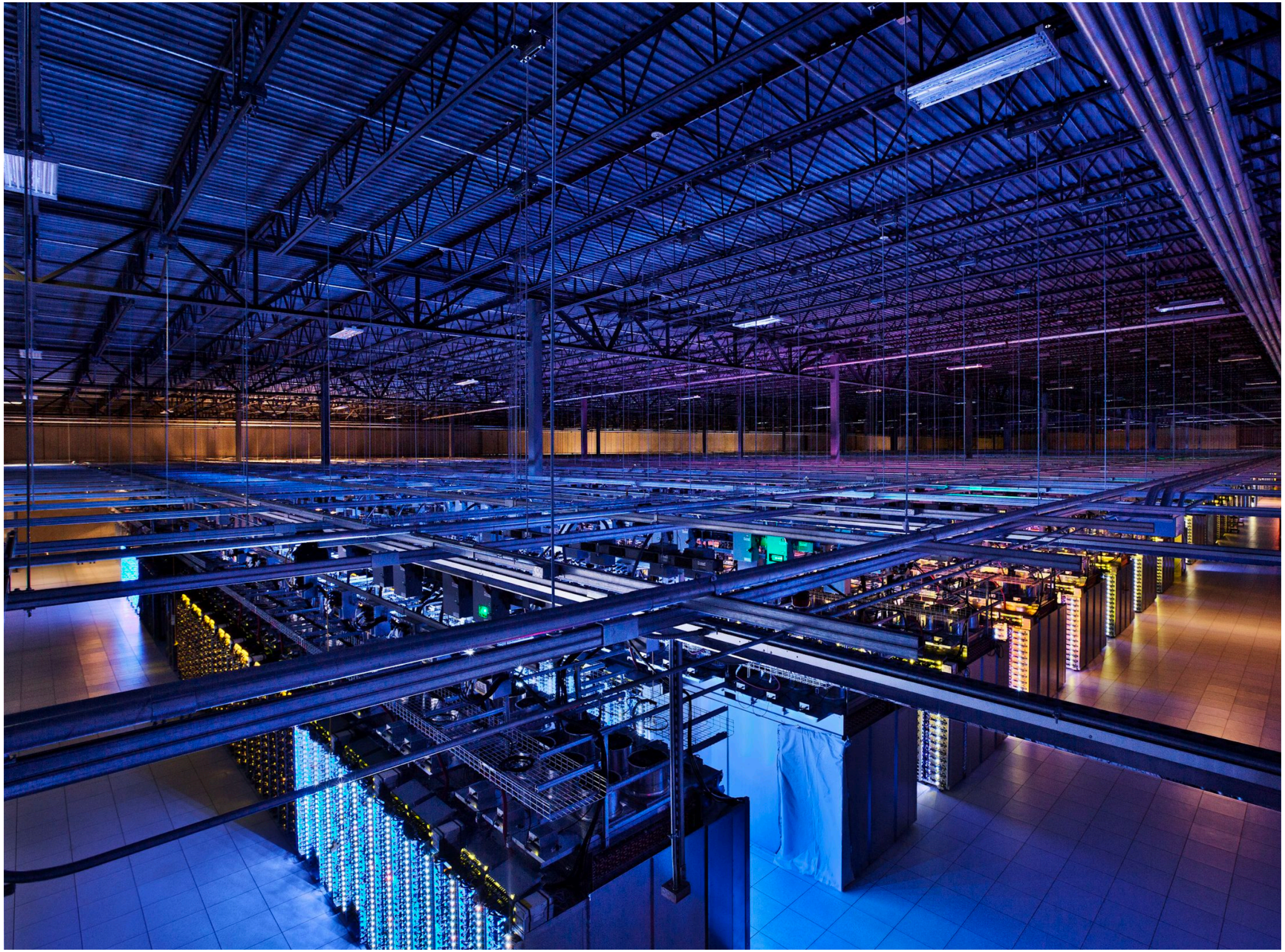


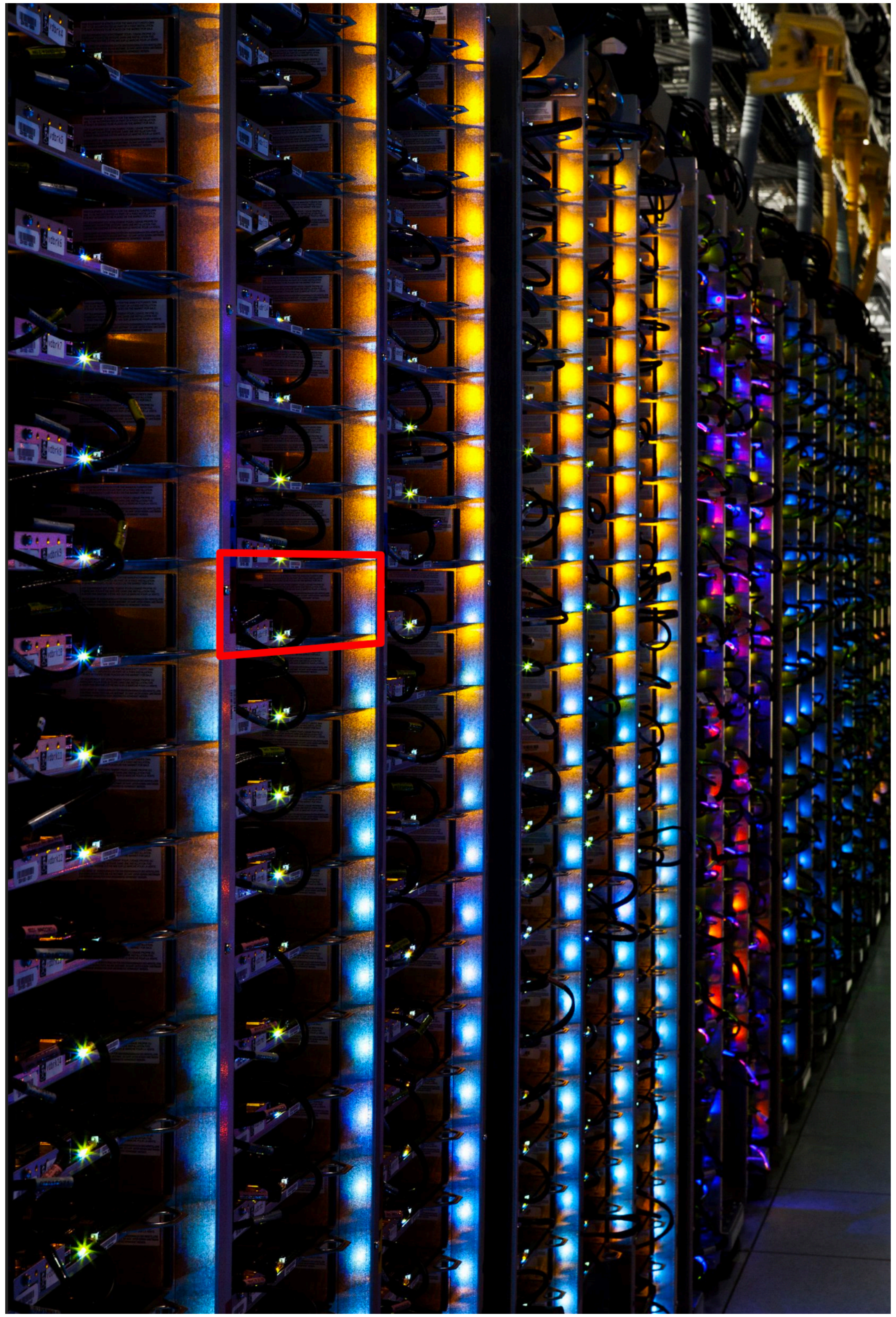
A TOUR TO DATACENTER

- 1. Datacenter hardware
- 2. Datacenter software
  - a. Google
  - b. Meta and Open source
  - c. Moving forward: ML stack

Hardware

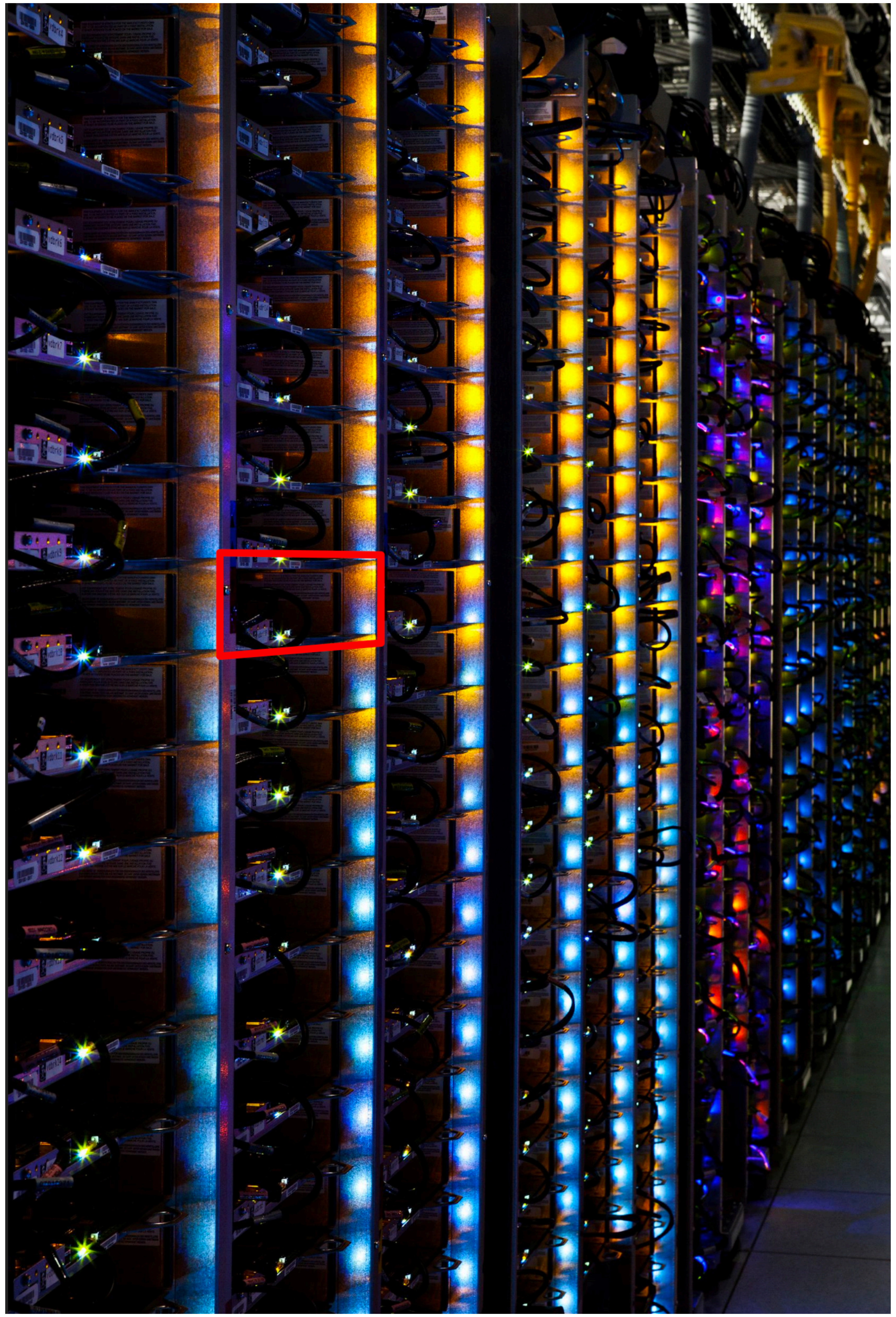

Amazon Web Services - IAD28
4.7 (65) · Corporate office in Fairfax County, Virginia
137+ Photos
Directions
Save
Call
Overview
Reviews
Address: 13200 Woodland Park Rd, Herndon, VA 20171
Phone: (571) 260-2603
Suggest an edit · Own this business?
Add missing information
Add business hours
Add website
Germantown
Poolesville
Gaithersburg
Leesburg
Potomac River
Rockville
Waxpool
Potomac
Aldie
Brambleton
Amazon Web Services - IAD28
Dulles
CoreSite Reston Data Center (VA1)
Element Critical Center - Norther
Chantilly
Data Center
Arlingto
Centreville
Fairfax
aymarket
Gainesville
Iron Mountain Data Centers VA-1
Burke
Amazon L
Manassas
Bristow
Fort Belvoir
Lorton
Catlett
Woodbridge
Dale City
Montclair
Prince William





From Meta (as of 2022):

– O(?) machines in total
– O(?) regions
– O(?) interdependent services
– “Machine”
  – no chassis
  – DC battery
  – mostly custom-made
– Network
  – ToR switch
  – multi-path core



From Meta (as of 2022):

- O(1M) machines in total
- O(10s) regions
- O(1000s) interdependent services
- "Machine"
  - no chassis
  - DC battery
  - mostly custom-made
- Network
  - ToR switch
  - multi-path core

THE JOYS OF REAL HARDWARE

Source: Jeff Dean https://static.googleusercontent.com/media/research.google.com/en//people/jeff/stanford-295-talk.pdf, 2007.

Typical first year for a new cluster:

~0.5 overheating (power down most machines in <5 mins, ~1-2 days to recover)

~1 PDU failure (~500-1000 machines suddenly disappear, ~6 hours to come back)

~1 rack-move (plenty of warning, ~500-1000 machines powered down, ~6 hours)

~1 network rewiring (rolling ~5% of machines down over 2-day span)

~20 rack failures (40-80 machines instantly disappear, 1-6 hours to get back)

~5 racks go wonky (40-80 machines see 50% packetloss)

~8 network maintenances (4 might cause ~30-minute random connectivity losses)

~12 router reloads (takes out DNS and external vips for a couple minutes)

~3 router failures (have to immediately pull traffic for an hour)

~dozens of minor 30-second blips for dns

~1000 individual machine failures

~thousands of hard drive failures

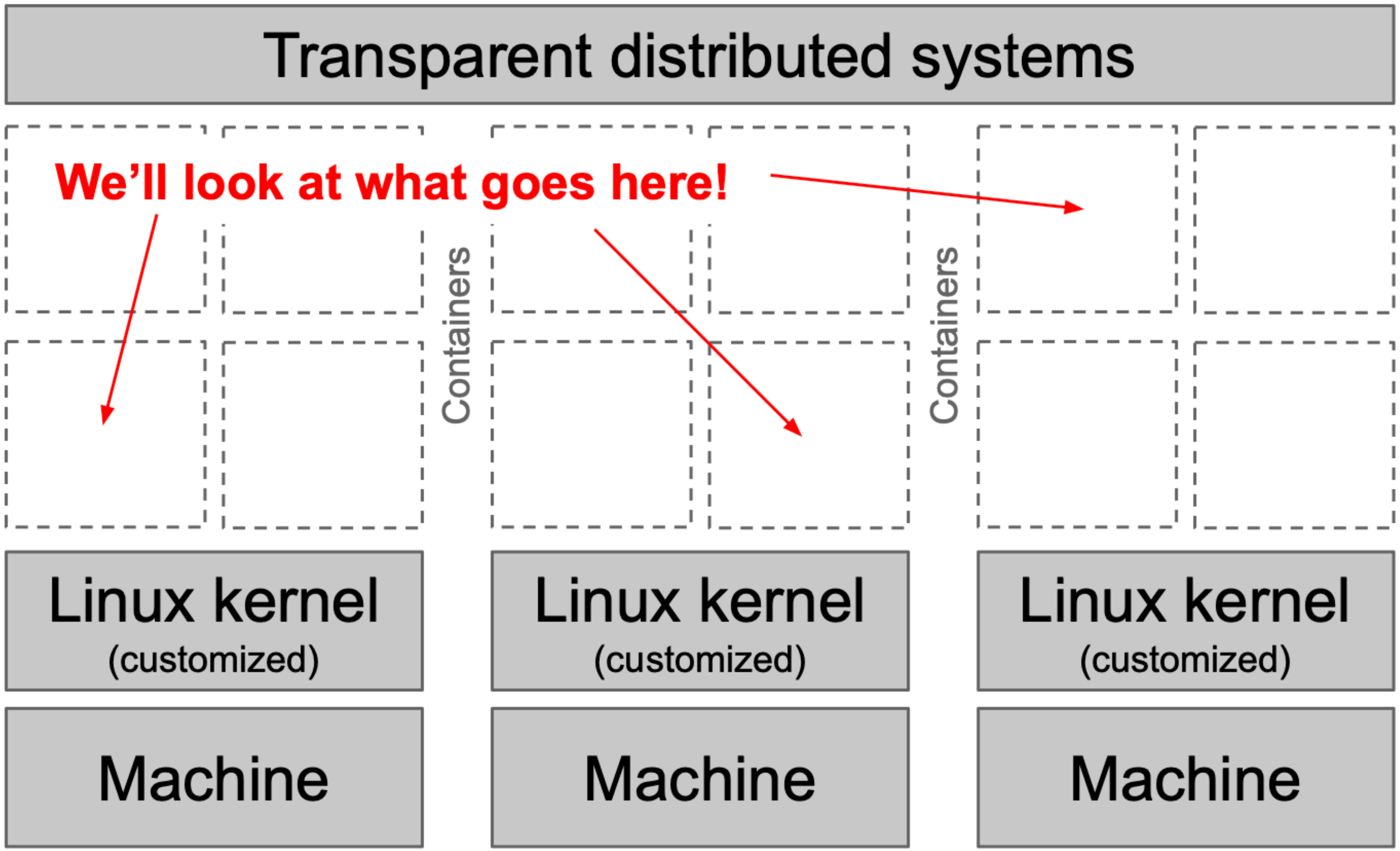
slow disks, bad memory, misconfigured machines, flaky machines, etc.

WHAT IT TAKES TO MANAGE LARGE-SCALE SYSTEMS
and how is it different from HPC?

- Emphasis on **commodity** hardware
  - No expensive interconnect
  - Mid-range machines
  - Energy/performance/cost trade-off essential
- Massive **automation**
  - Very small number of on-site staff
  - Automated software bootstrap
- **Fault tolerant** design
  - Each component can fail
  - Software must be aware and compensate

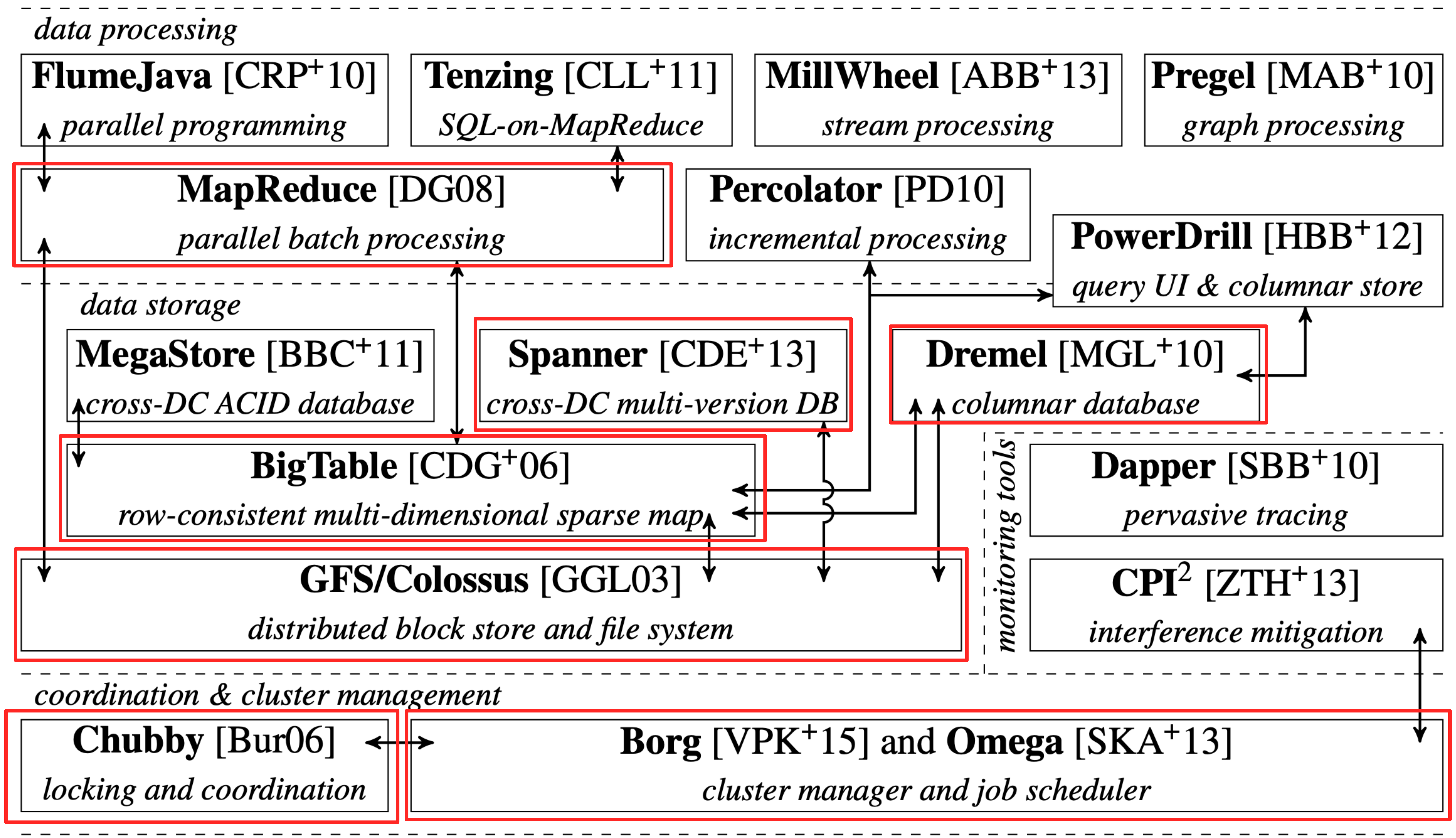
Software

SOFTWARE SYSTEMS STACK



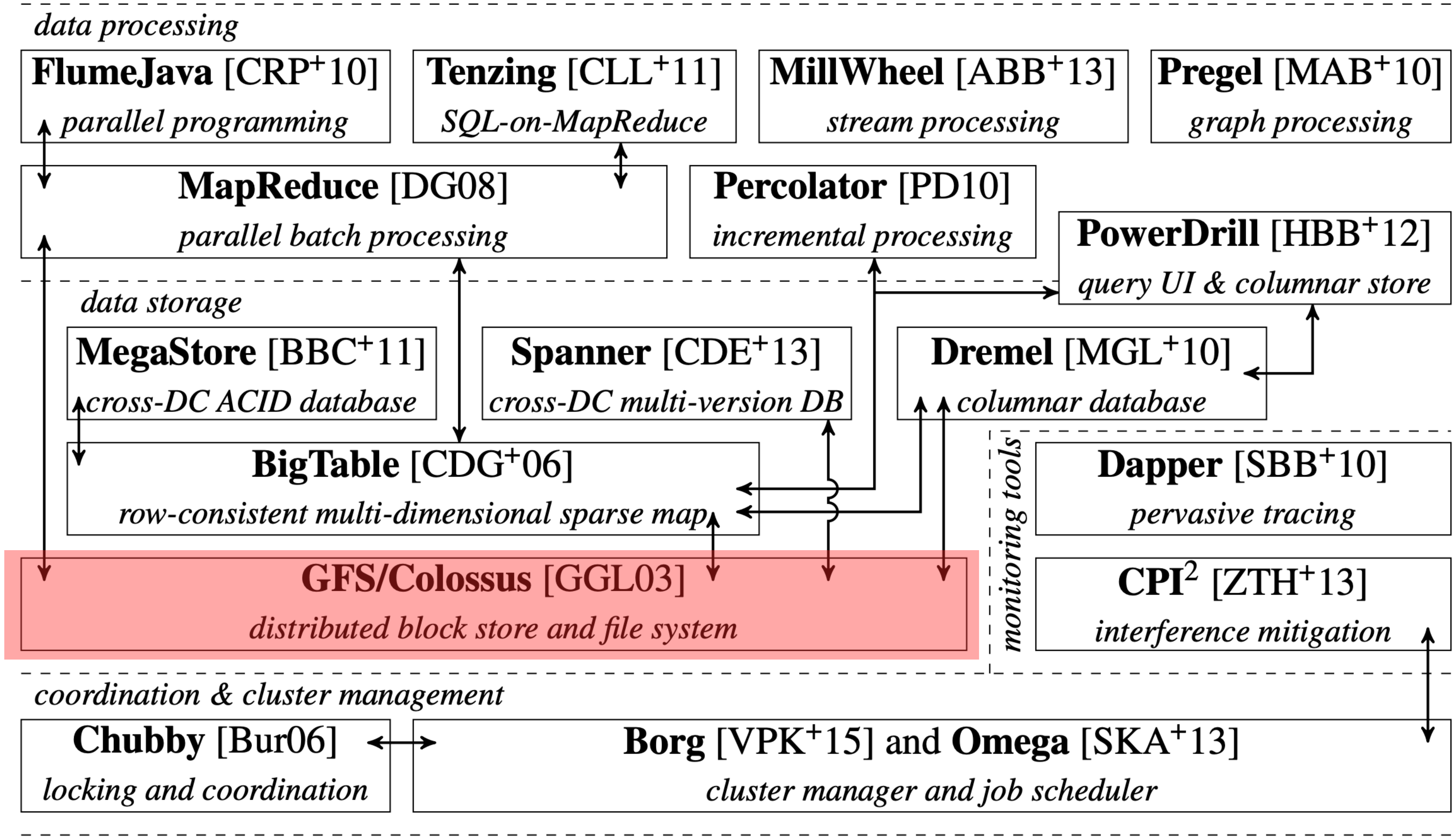
THE Google STACK

https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



THE Google STACK

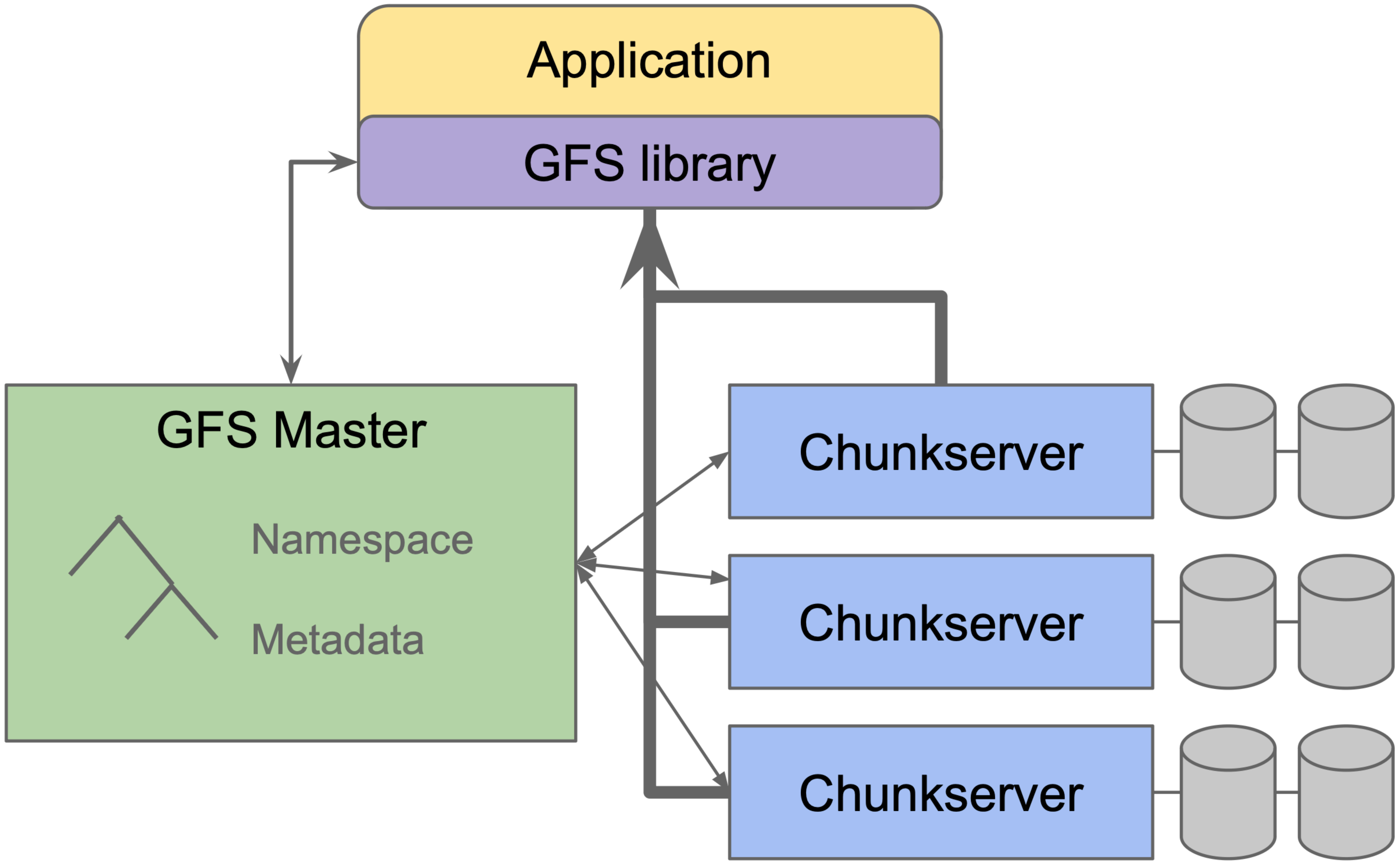
https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



GFS/COLOSSUS

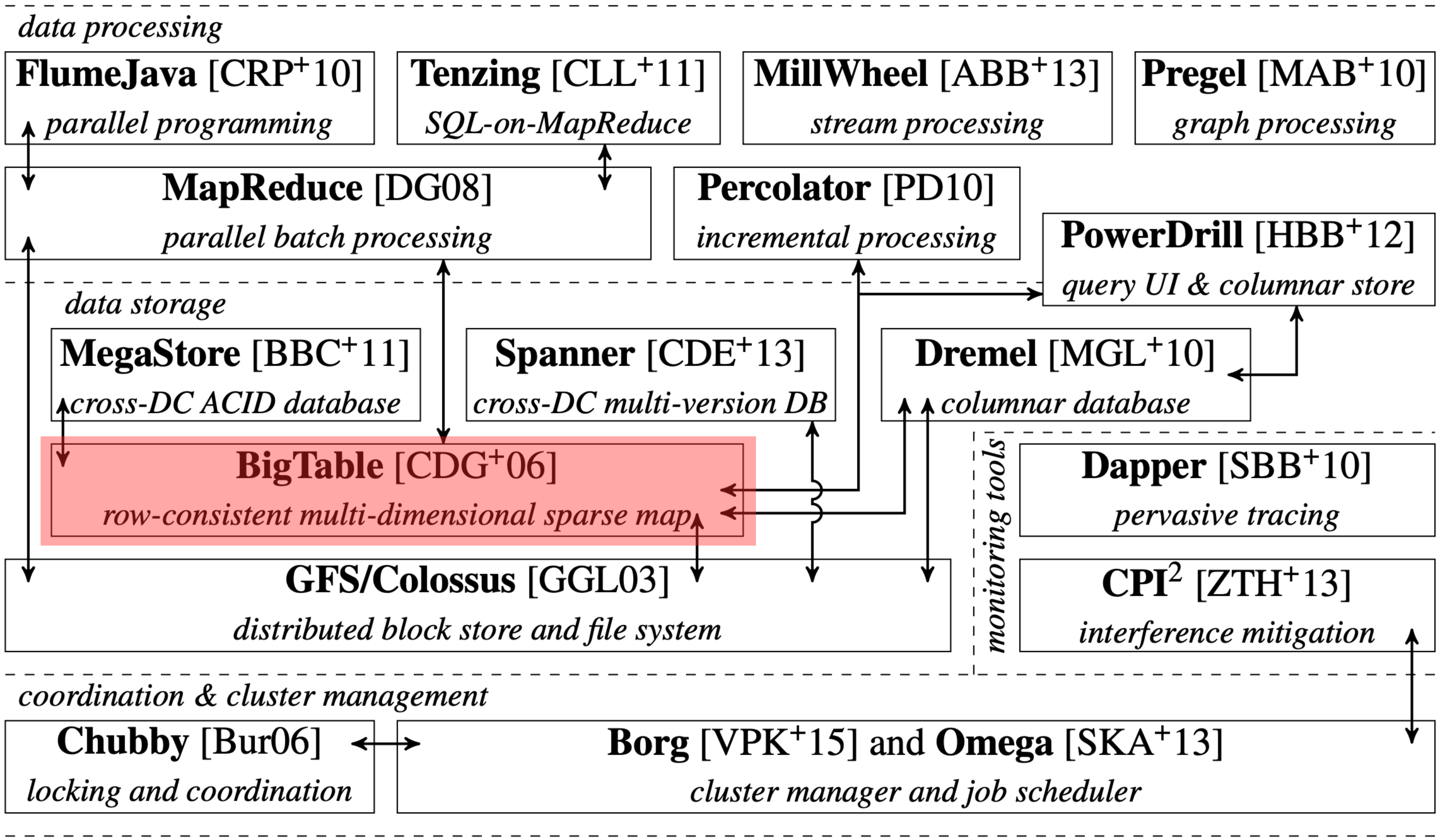
- Bulk data block storage system
  - Optimized for large files (GB-size)
  - Supports small files, but not common case
  - Read, write, record-append modes
- Colossus = GFSv2, adds some improvements
  - e.g., Reed-Solomon-based erasure coding
  - better support for latency-sensitive applications
  - sharded meta-data layer, rather than single master

GFS/COLOSSUS



THE Google STACK

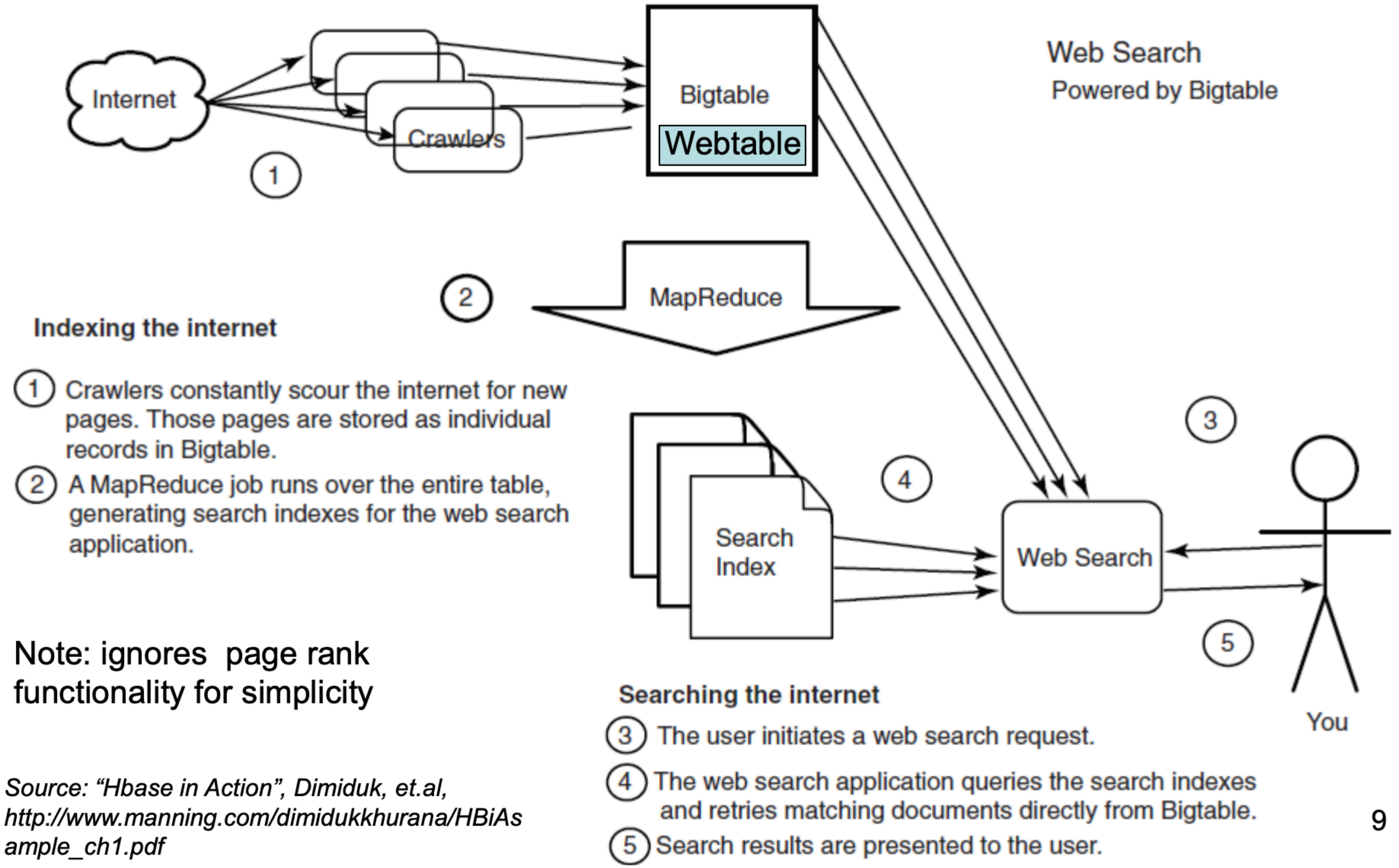
https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



MOTIVATION

- Lots of (semi-)structured data at Google
  - Web data: Contents, crawl metadata, links, anchors, pagerank, …
  - Per-user data: • User preference settings, recent queries/search results, …
  - Map data: • Physical entities (shops, restaurants, etc.), roads, satellite image data, user annotations, …
- Scale is huge
  - Billions of Web pages, many versions/page (~20K/version)
  - Hundreds of millions of users, thousands of q/sec
  - 100TB+ of satellite image data
  - (Above numbers are as of 2006-7!)

WEB SEARCH: THE COMPLETE WORKFLOW



Note: ignores page rank functionality for simplicity

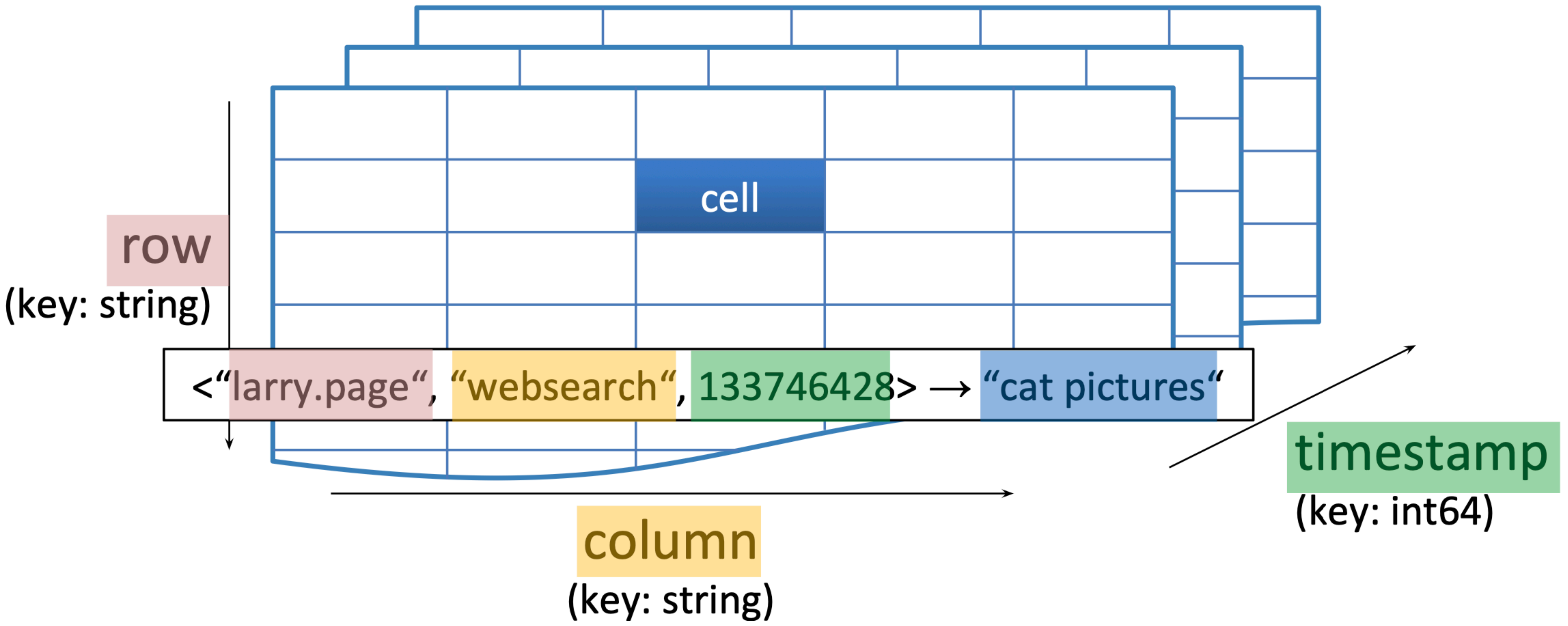
Source: "Hbase in Action", Dimiduk, et.al, http://www.manning.com/dimidukkhurana/HBiAsample_ch1.pdf

GOALS

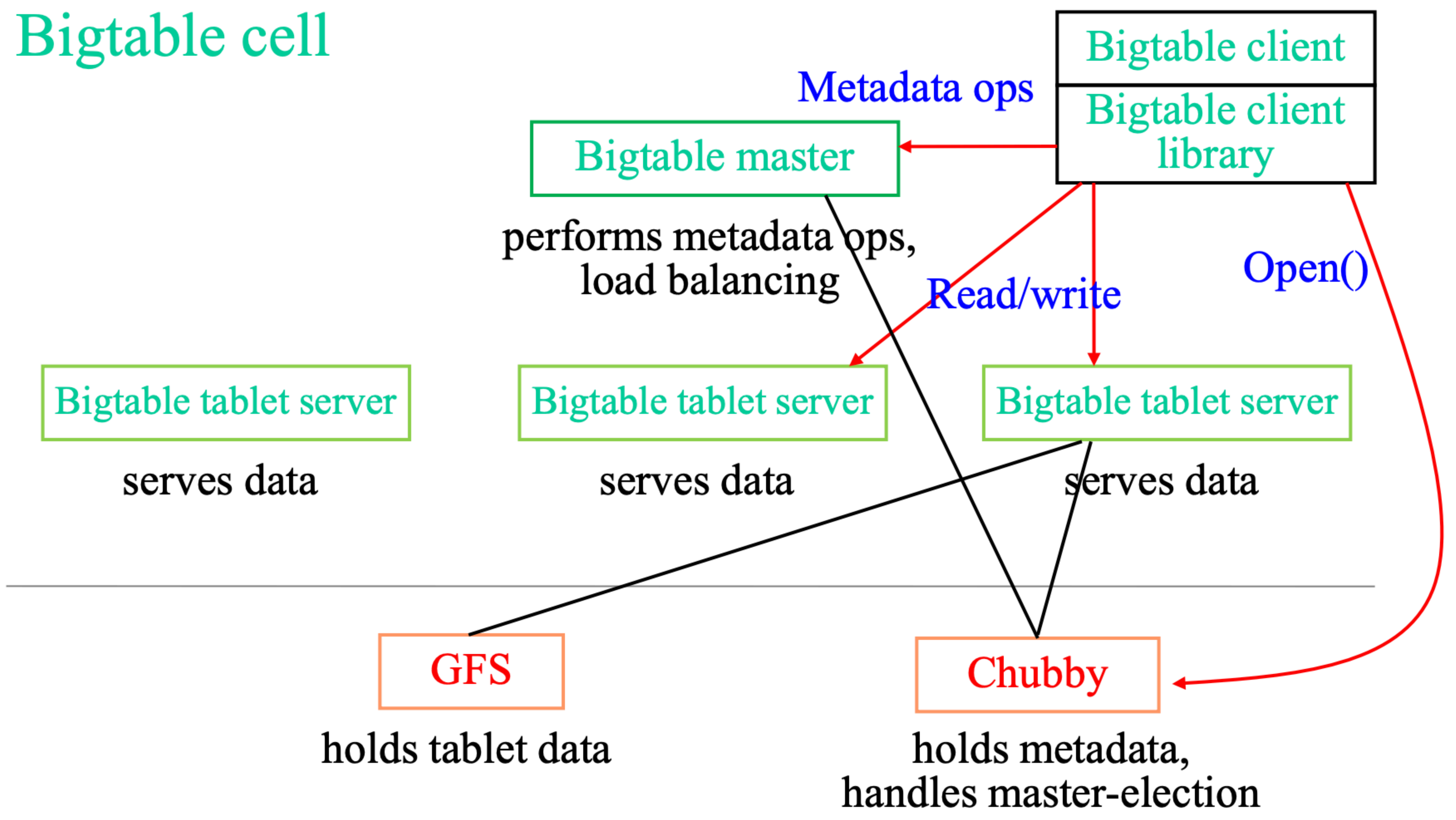
- Want asynchronous processes to continuously update different pieces of data
  - Want access to most current data at any time
- Need to support:
  - Very high read/write rates (millions of ops per second)
  - Efficient retrieval of small subsets of the data
  - Efficient scans over entire or subsets of the data
- Often want to examine data changes over time
  - E.g. Contents of a web page over multiple crawls

BIGTABLE (2006)

- 'Three-dimensional' key-value store:
  - <row key, column key, timestamp> → value
- Effectively a distributed, sorted, sparse map

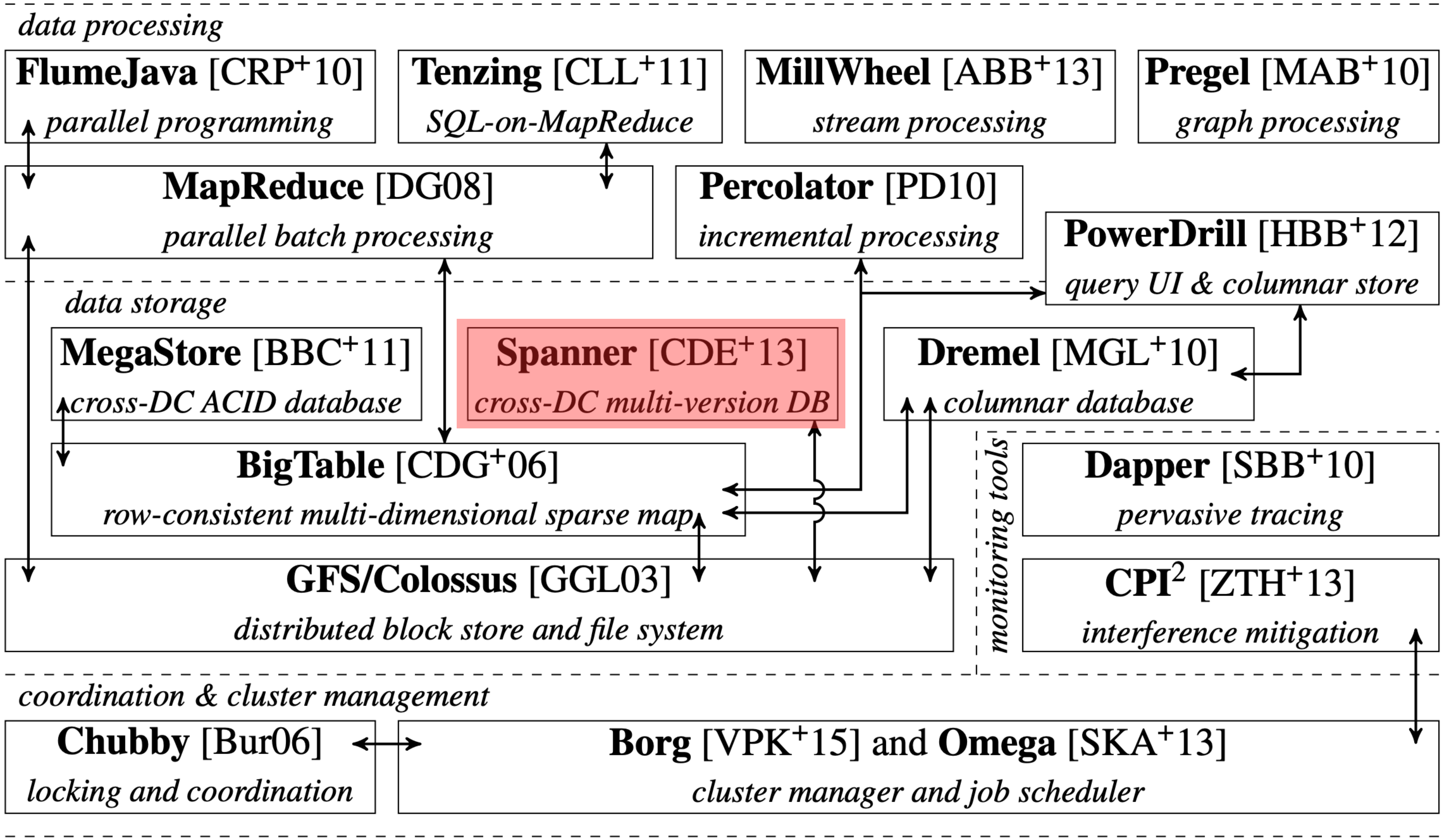


SYSTEM ARCHITECTURE



THE Google STACK

https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf

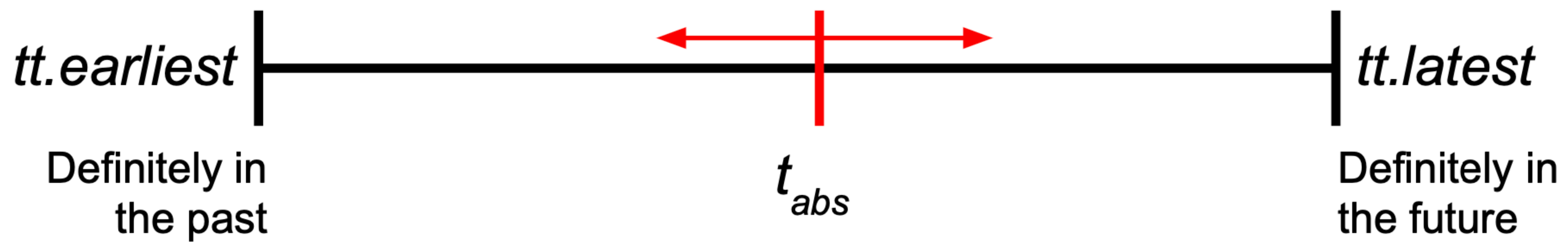


SPANNER (2012)

- BigTable insufficient for some consistency needs
- Often have transactions across >1 data centers
  - May buy app on Play Store while travelling in the U.S.
  - Hit U.S. server, but customer billing data is in U.K.
  - Or may need to update several replicas for fault tolerance
- Wide-area consistency is hard
  - due to long delays and clock skew
  - no global, universal notion of time
  - NTP not accurate enough, PTP doesn’t work (jittery links)

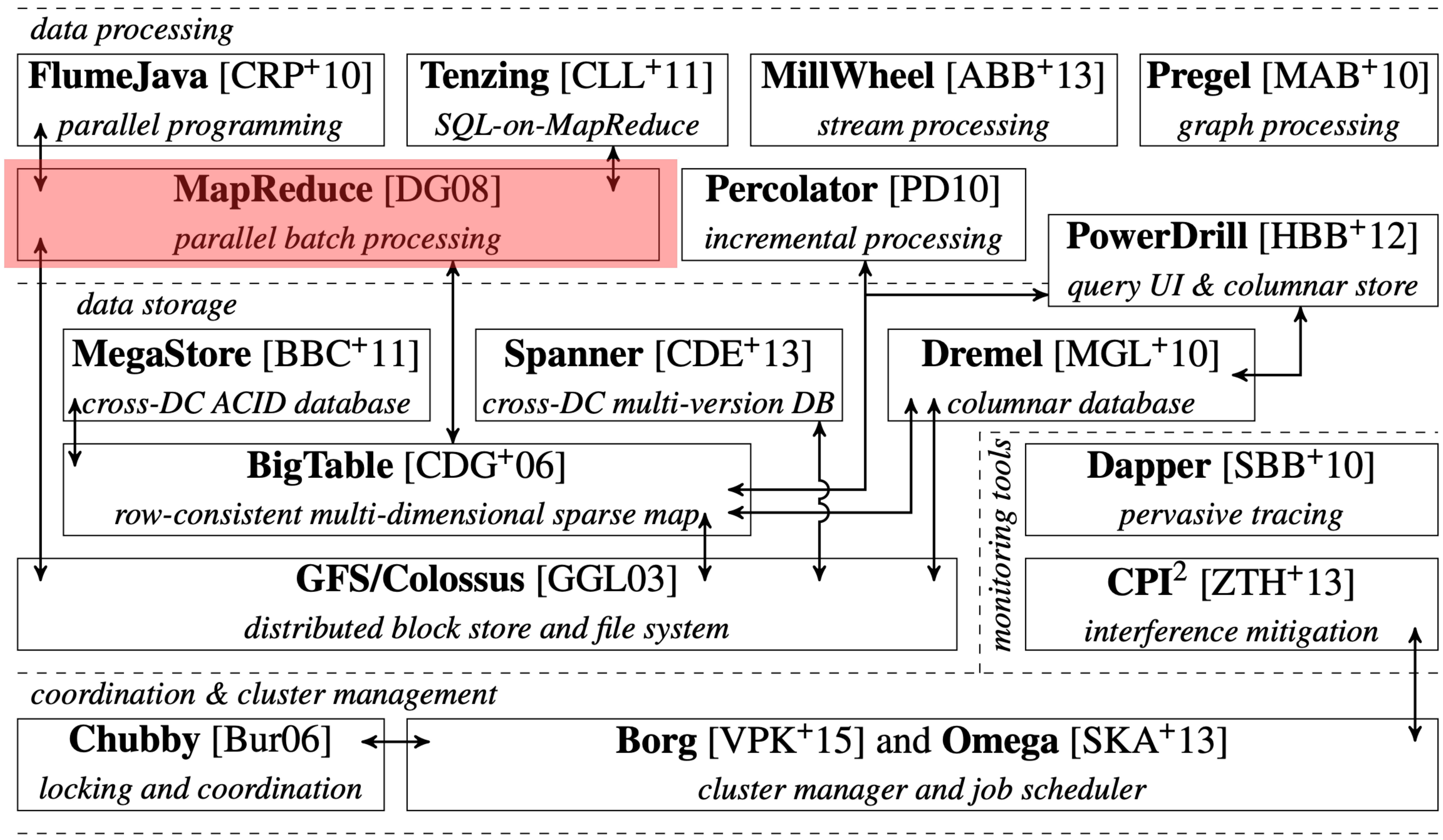
SPANNER (2012)

- Spanner offers transactional consistency: full RDBMS
- Secret sauce: hardware-assisted clock sync
  - Using GPS and atomic clocks in data centres
- Use global timestamps and Paxos to reach consensus
  - Still have a period of uncertainty for write TX: wait it out!
  - Each timestamp is an interval:



THE Google STACK

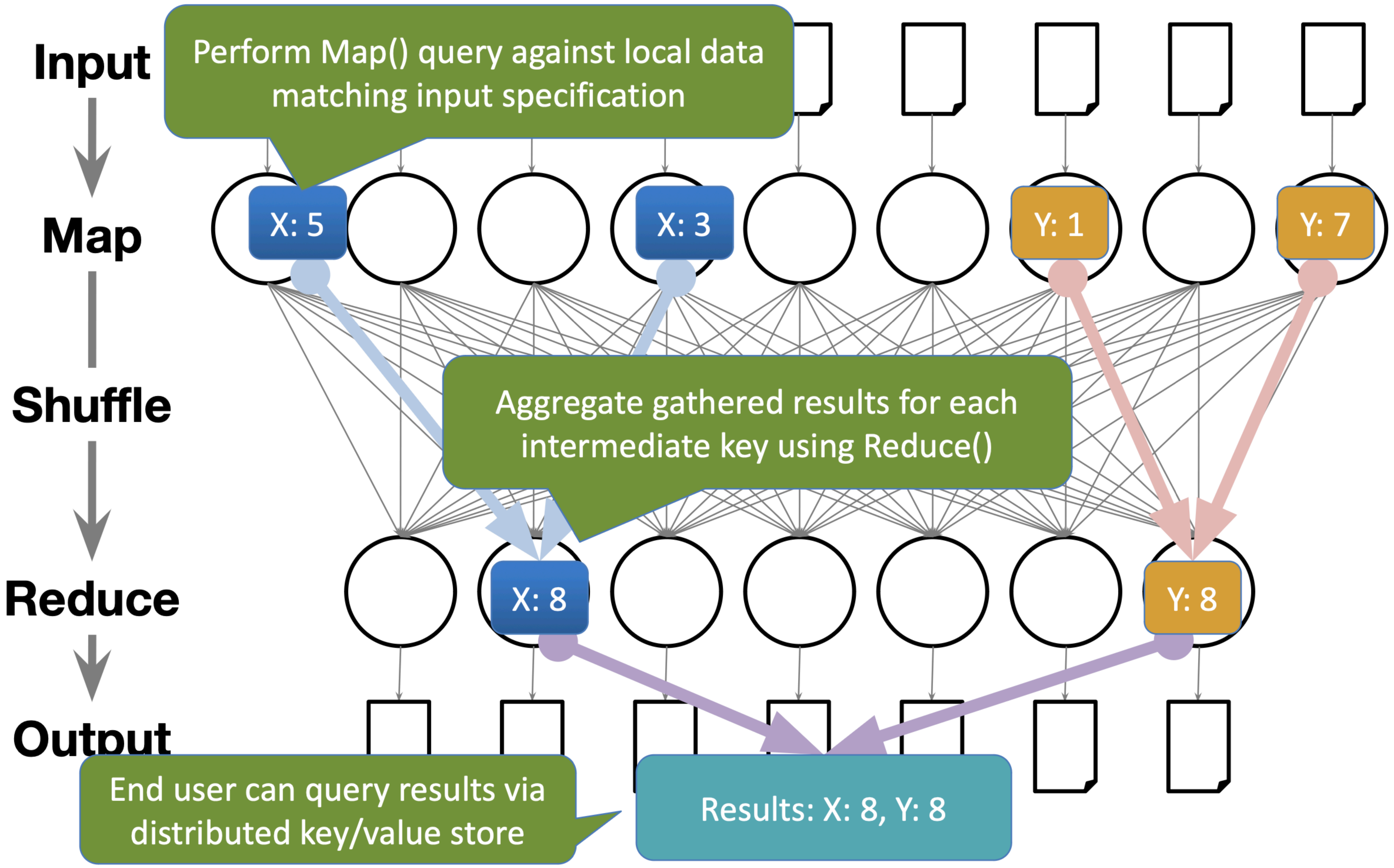
https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



MAPREDUCE (2004)

- Parallel programming framework for scale
  - Run a program on 100’s to 10,000’s machines
- Framework takes care of:
  - Parallelization, distribution, load-balancing, scaling up (or down) & fault-tolerance
- Accessible: programmer provides two methods ;-)
  - map(key, value) → list of <key’, value’> pairs
  - reduce(key’, value’) → result
  - Inspired by functional programming

MAPREDUCE (2004)

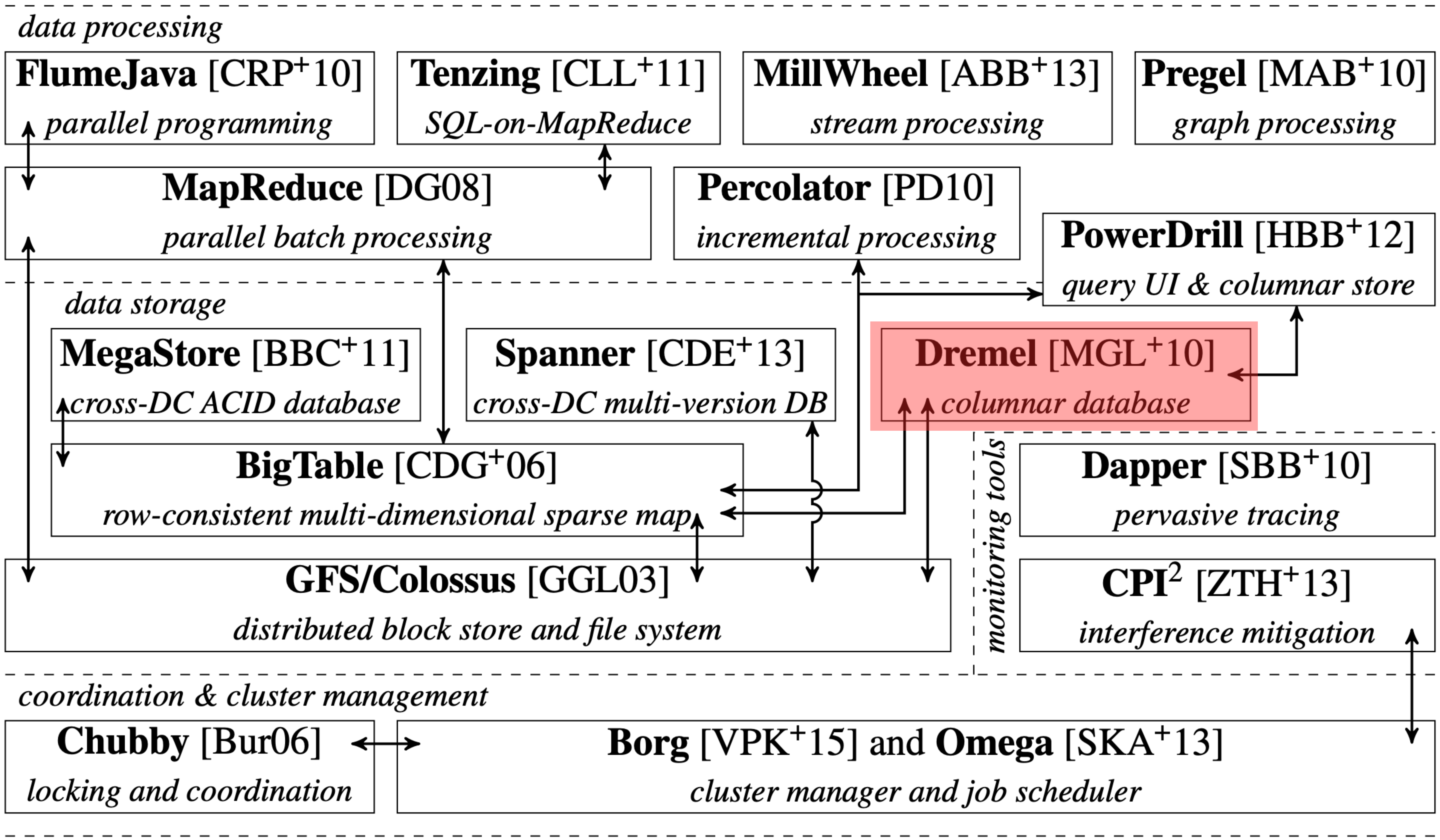


MAPREDUCE: PROS & CONS

- Extremely simple, and:
  - Can auto-parallelize (since operations on every element in input are independent)
  - Can auto-distribute (since rely on underlying Colossus/BigTable distributed storage)
  - Gets fault-tolerance (since tasks are idempotent, i.e. can just re-execute if a machine crashes)
- Doesn't really use any sophisticated distributed systems algorithms (except storage replication)
- However, not a panacea:
  - Limited to batch jobs, and computations which are expressible as a map() followed by a reduce()

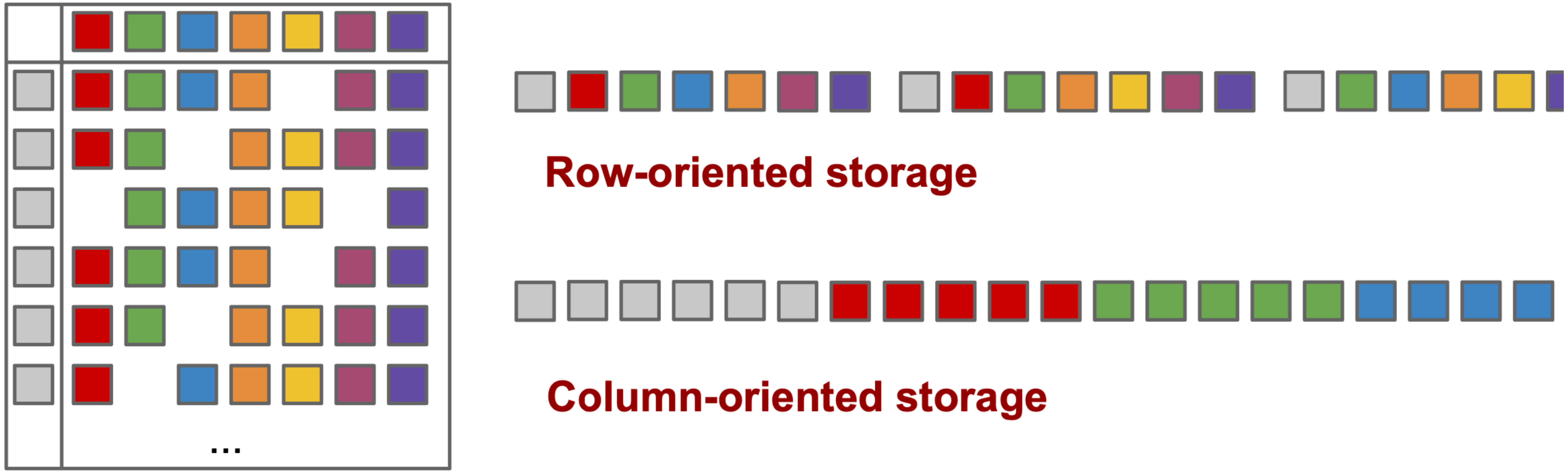
THE Google STACK

https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



DREMEL (2010)

- Column-oriented store
  - For quick, interactive queries



DREMEL (2010)

user_id	user_name	current_balance	number_of_transactions
1	'freddie@gmail.com'	1059298	1224
2	'lindsey@gmail.com'	254	1045
3	'tabby@yahoo.com'	3910	194
4	'philip@hotmail.com'	234028	130
5	'elon@x.com'	-44000000000	1

```
SELECT sum(current_balance)
FROM table
WHERE user_id > 2
```

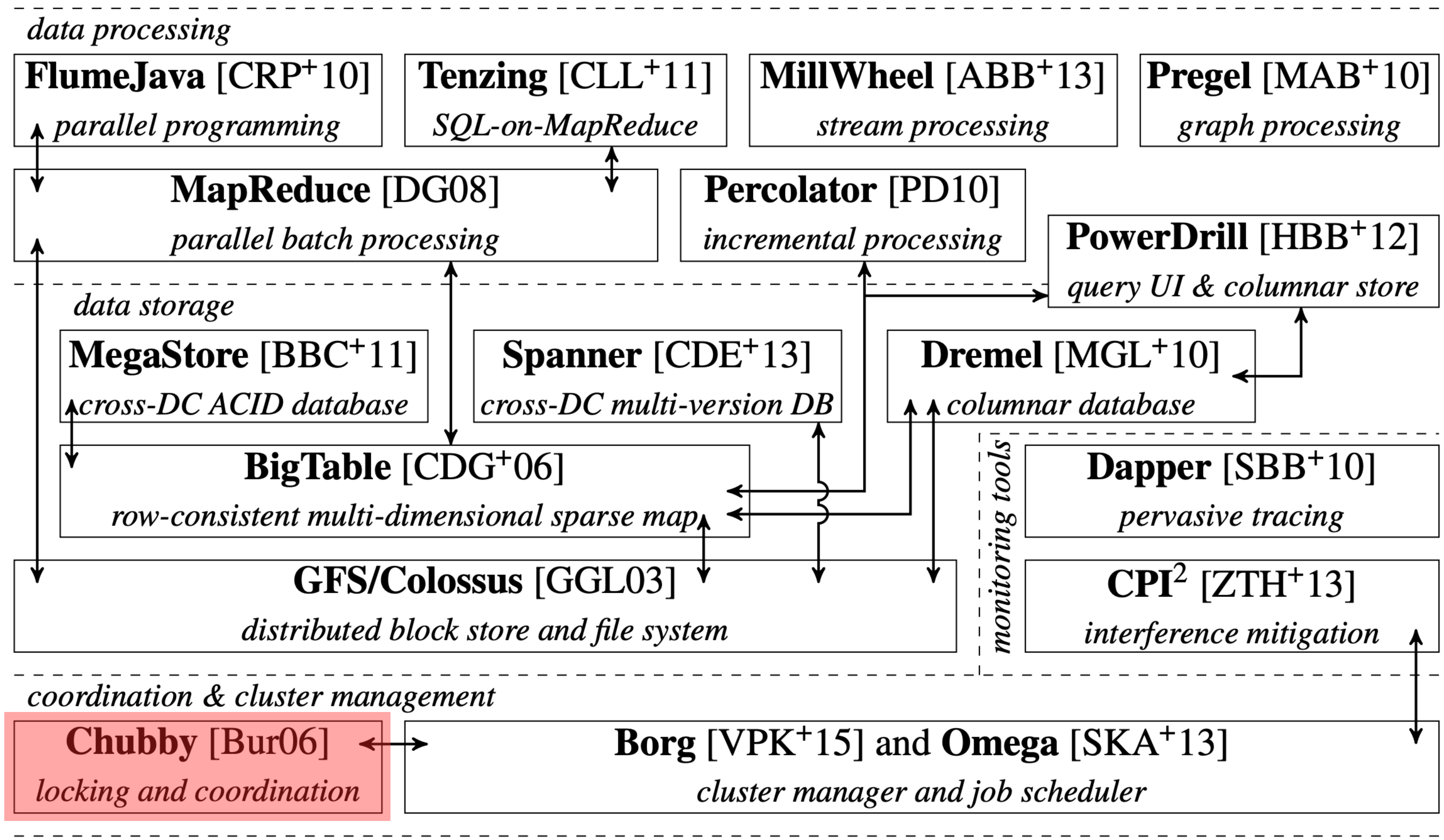
Suitable for columnar DB

```
SELECT user_id, user_name,
       current_balance
FROM table
WHERE user_id = 1
```

Suitable for row-oriented DB

THE Google STACK

https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



CHUBBY SUMMARY

- Lock Service
- Chubby uses Paxos for everything
  - Propagate writes to a file
  - Choosing a Master
  - Even for adding new Chubby servers to a Chubby cell
- Paxos transforms a multi-node service into something that looks very much like one fault-tolerant, albeit slower, server! -> pretty close to distributed systems' core goal

CHUBBY INTERFACE: UNIX FILE SYSTEM

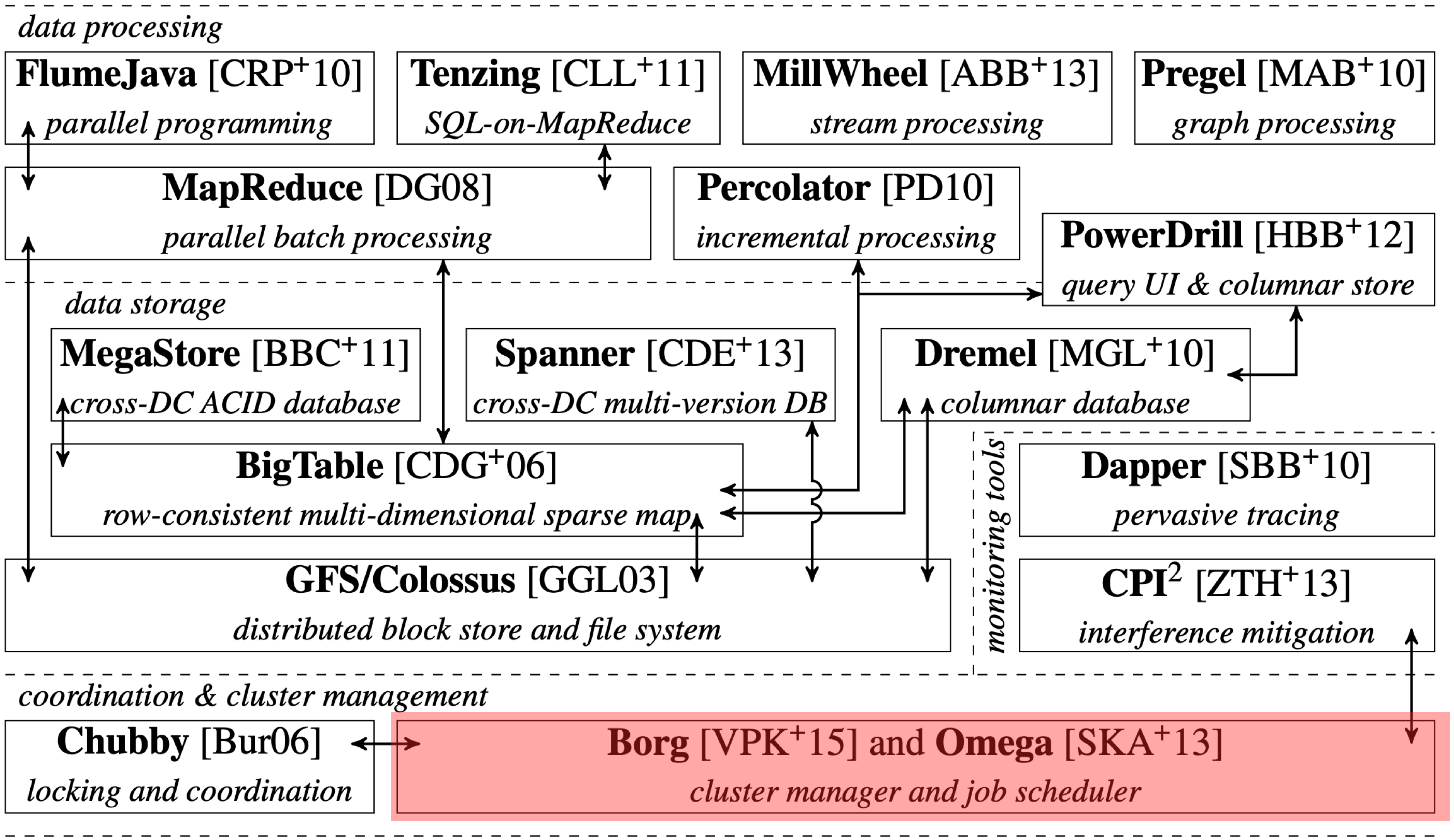
- Chubby supports a strict tree of files and directories
  - The way to think about these files is that they are locks with a little bit of contents (e.g., identity and location of a primary)
  - No symbolic links, no hard links
  - /ls/foo/wombat/pouch
    - 1st component (ls): lock service (common to all names)
    - 2nd component (foo): the chubby cell (used in DNS lookup to find the cell master)
    - The rest: name inside the cell
- Support most normal operations
  - Create, delete, open, write, ...
- Support reader/writer lock on a node

EXAMPLE: PRIMARY ELECTION

```
Open("/ls/foo/OurServicePrimary", "write mode");
if (successful) {
  // primary
  SetContents(primary_identity);
} else {
  // replica
  Open("/ls/foo/OurServicePrimary", "read mode",
    "file-modification event");
  when notified of file modification:
    primary = GetContentsAndStat();
}
```

THE Google STACK

https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



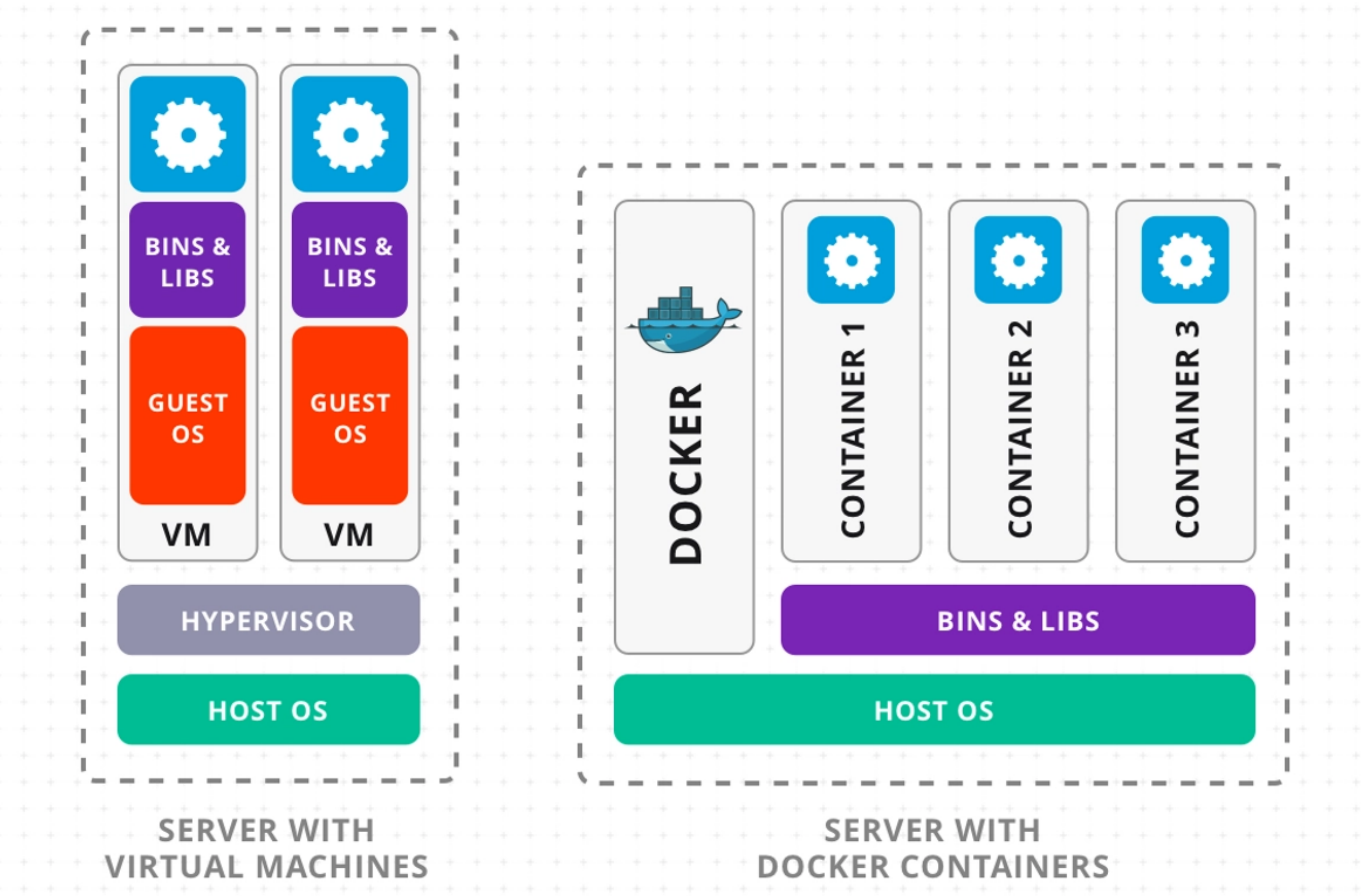
BORG

- Cluster manager and scheduler
  - Tracks machine and task liveness
  - Decides where to run what
- Consolidates workloads onto machines
  - Efficiency gain, cost savings
  - Need fewer clusters
- You might be more familiar with its successor:

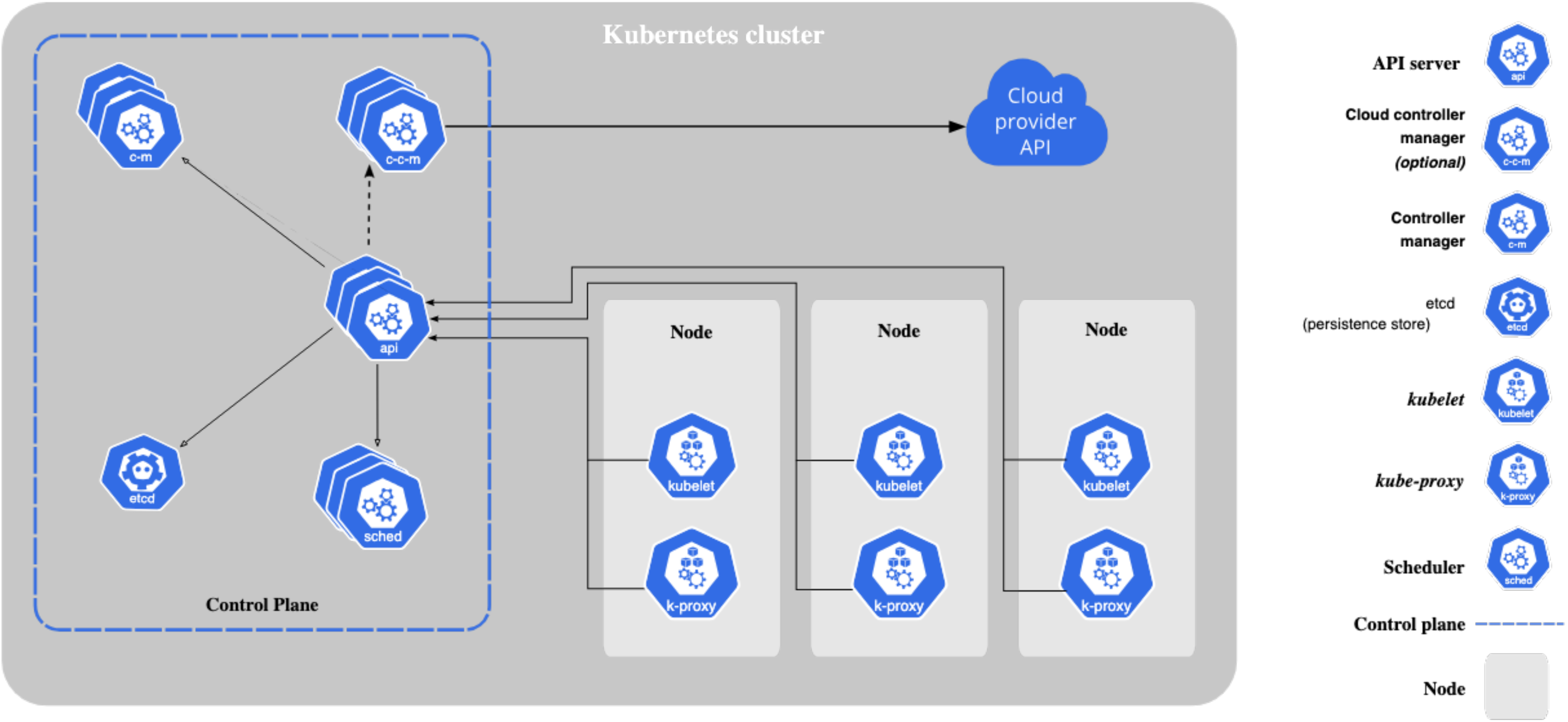
kubernetes

BACKGROUND: CONTAINERS

https://hanwenzhang123.medium.com/docker-vs-virtual-machine-vs-kubernetes-overview-389db7de7618

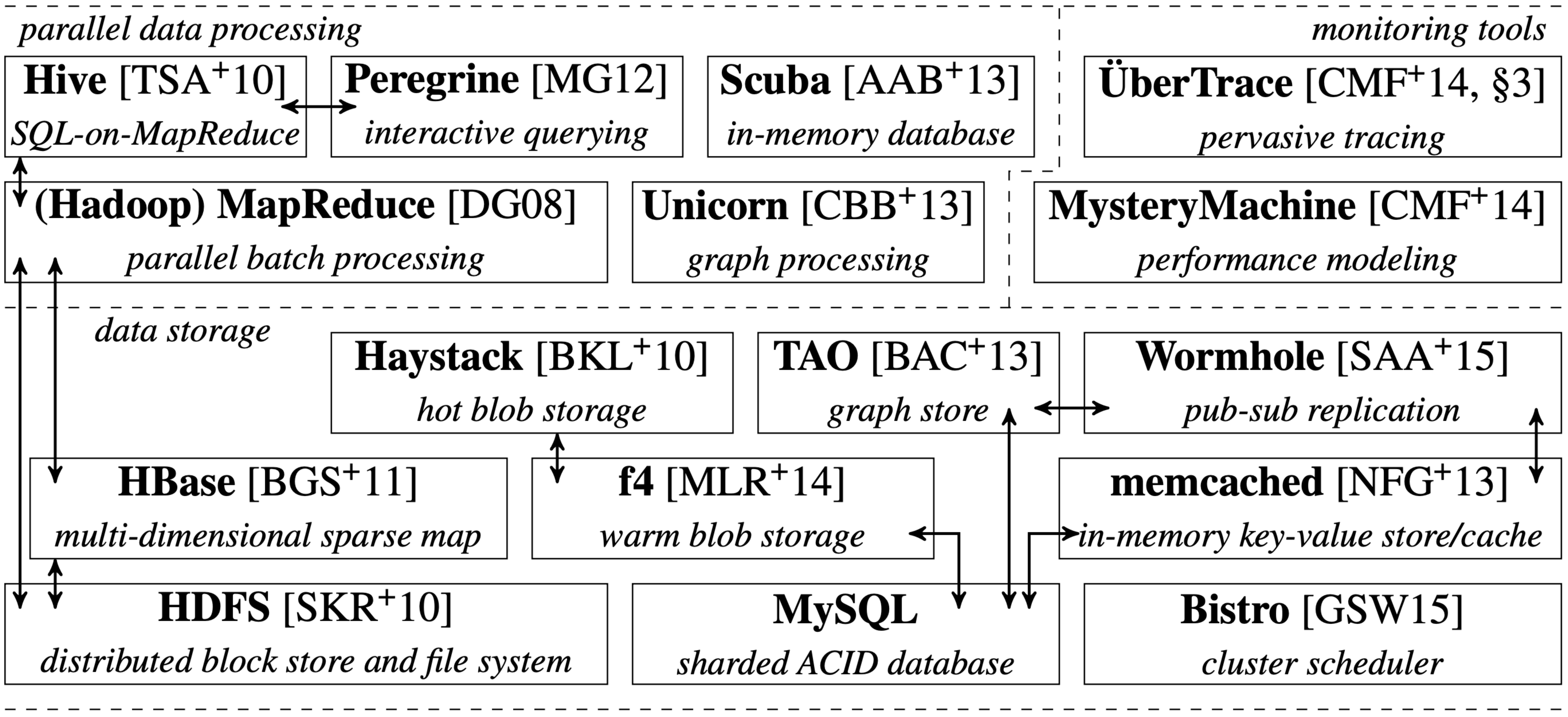


KUBERNETES ARCHITECTURE



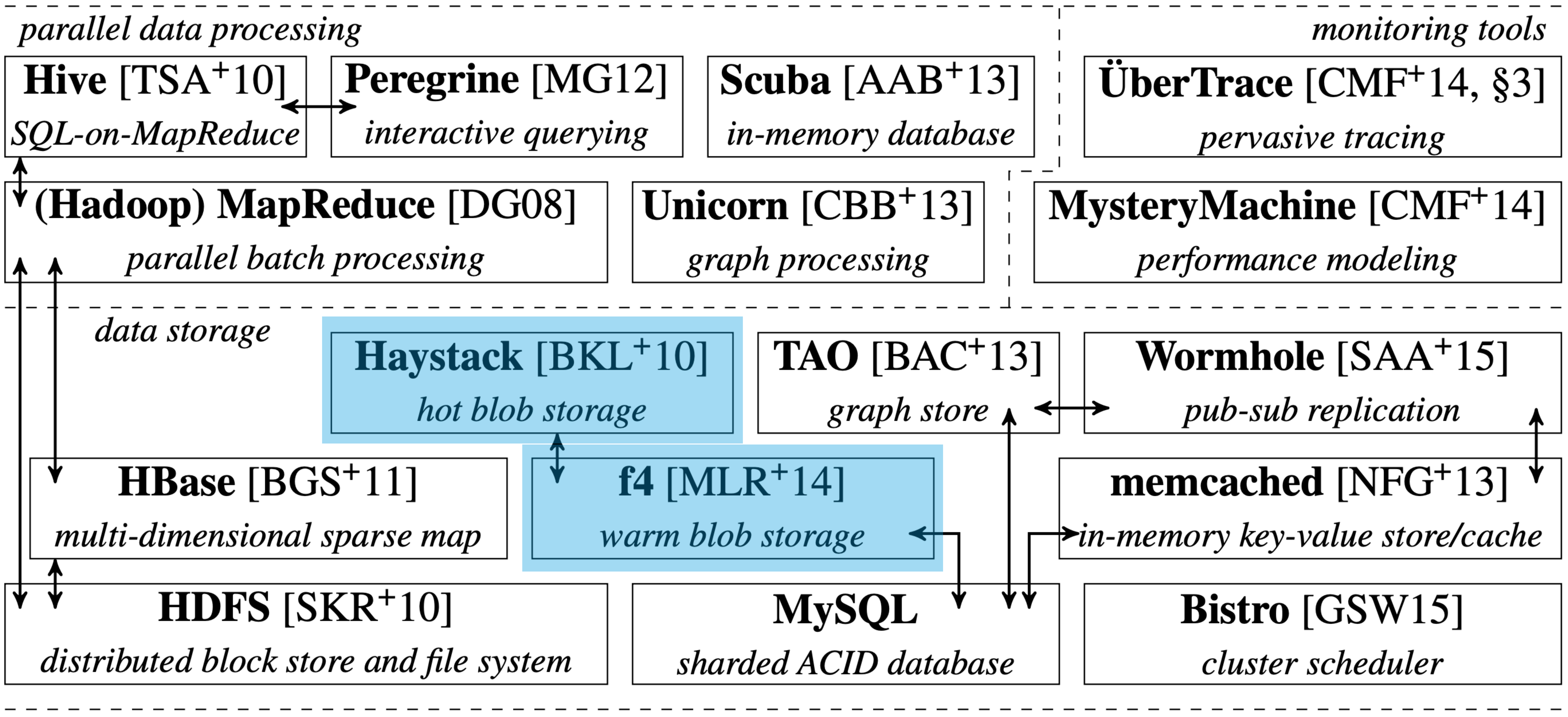
THE Meta STACK

https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



THE Meta STACK

https://people.csail.mit.edu/malte/pub/dissertations/phd-final.pdf



HAYSTACK & F4

- Blob stores, hold photos, videos
  - not: status updates, messages, like counts
- Items have a level of hotness
  - How many users are currently accessing this?
  - Baseline "cold" storage: MySQL
- Want to cache close to users
  - Reduces network traffic
  - Reduces latency
  - But cache capacity is limited!
  - Replicate for performance, not resilience

What about other companies' stacks?

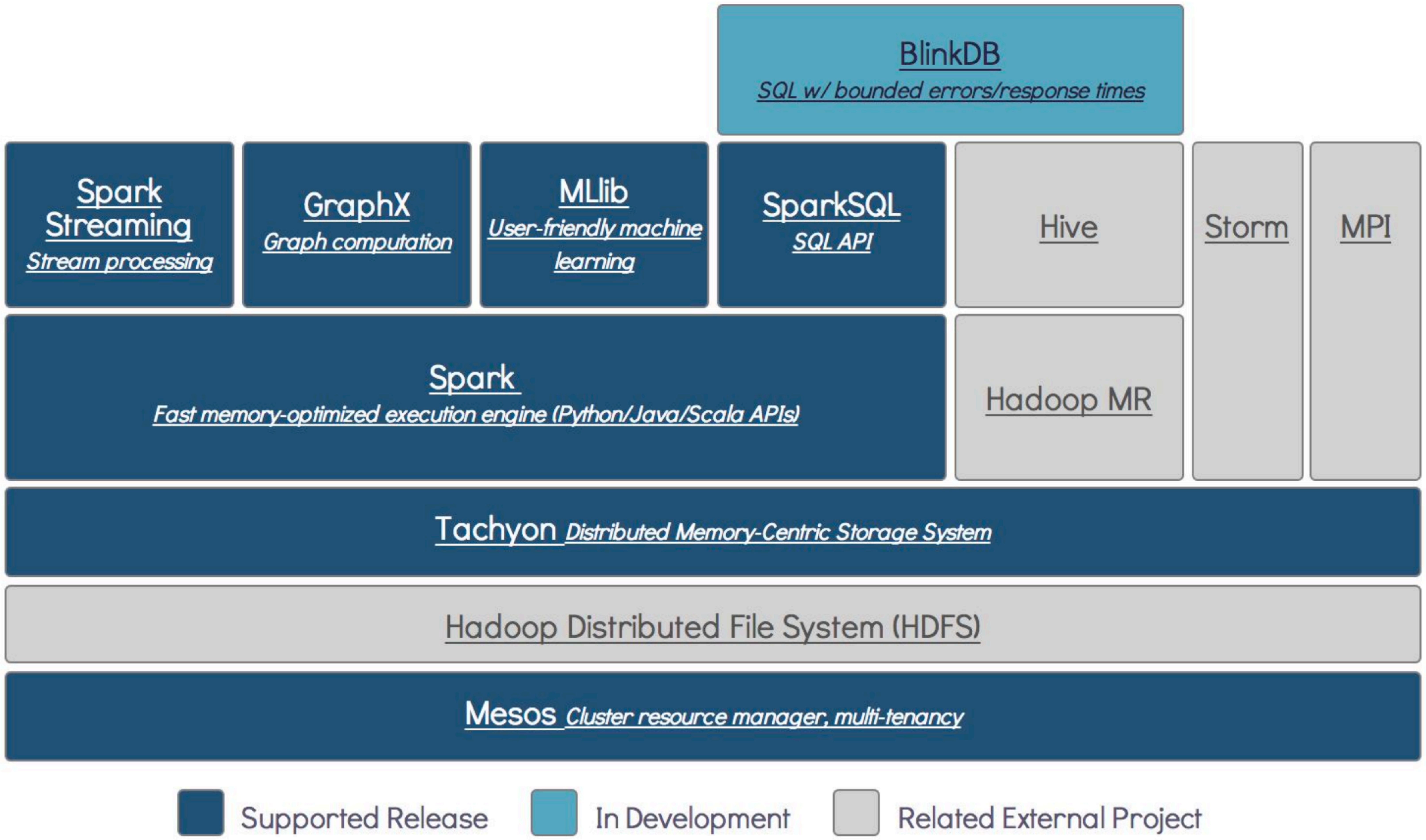
HOW ABOUT OTHER COMPANIES?

- Very similar stacks.
  - Microsoft, Yahoo, Twitter all similar in principle.
- Typical set-up:
  - Front-end serving systems and fast back-ends.
  - Batch data processing systems.
  - Multi-tier structured/unstructured storage hierarchy.
  - Coordination system and cluster scheduler.
- Minor differences owed to business focus
  - e.g., Amazon focused on inventory/shopping cart.

OPEN SOURCE SOFTWARE

- Lots of open-source implementations!
  - MapReduce → Hadoop, Spark, Metis
  - GFS → HDFS
  - BigTable → HBase, Cassandra
  - Borg → Mesos, Firmament
  - Chubby → Zookeeper
- But also some releases from companies...
  - Presto (Facebook)
  - Kubernetes (Google Borg)

THE Spark STACK

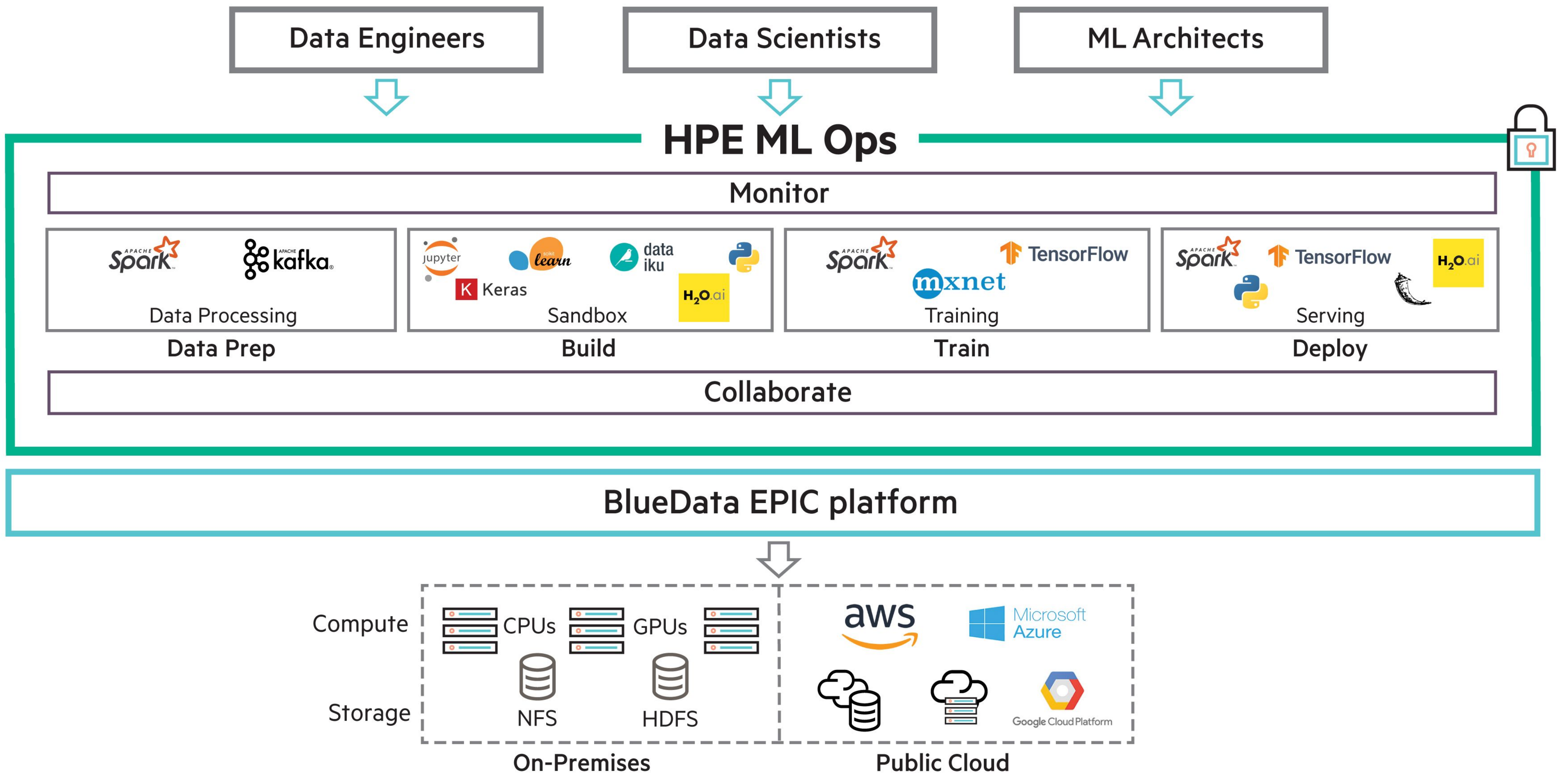


NEWER STACKS

- Lots of new support for machine learning
  - Google: Tensorflow, Tensorflow Serving, Tensorflow Extended (TFX)
  - Uber: Michelangelo
  - Spark/Berkeley Data Stack (BDAS): MLBase, MLlib, Clipper

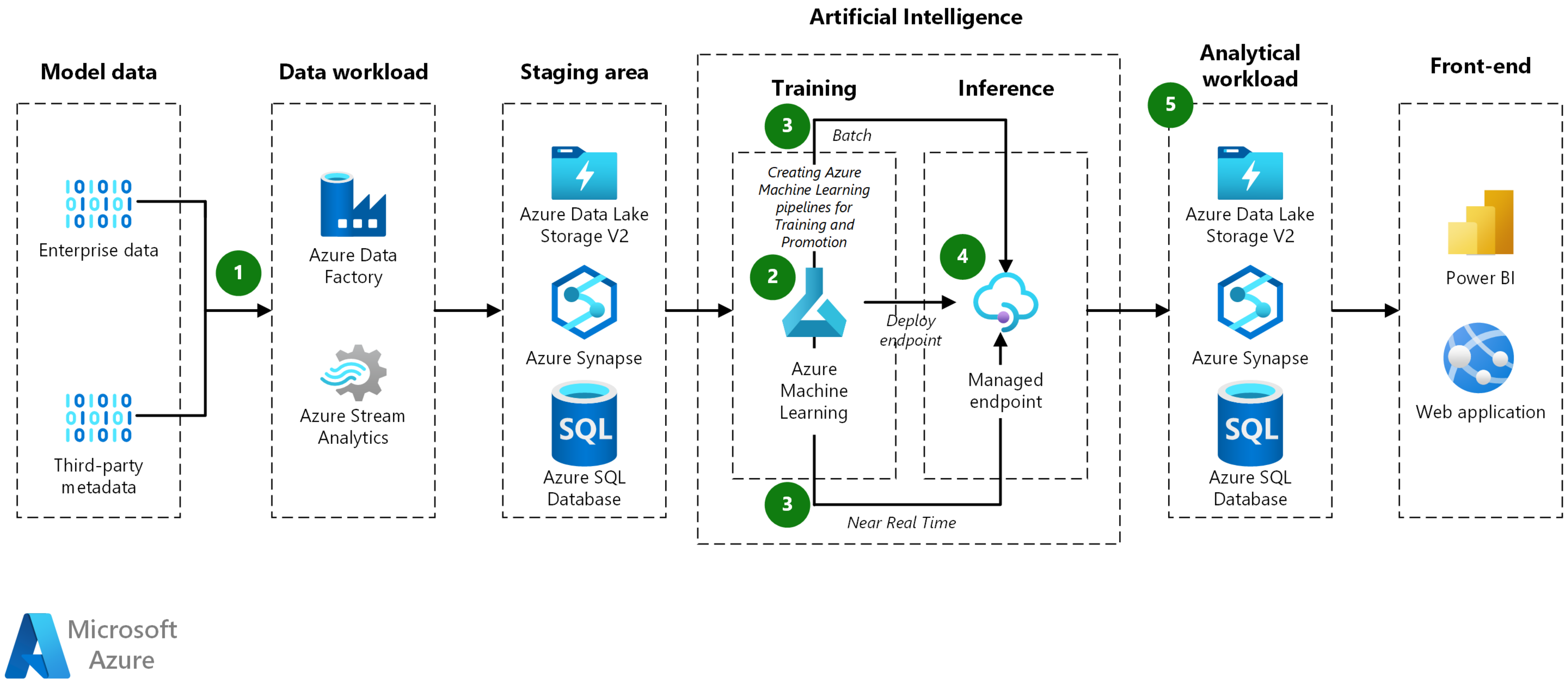
HEWLETT-PACKARD (HP)

https://community.hpe.com/t5/hpe-ezmeral-uncut/machine-learning-operationalization-in-the-enterprise/ba-p/7062451



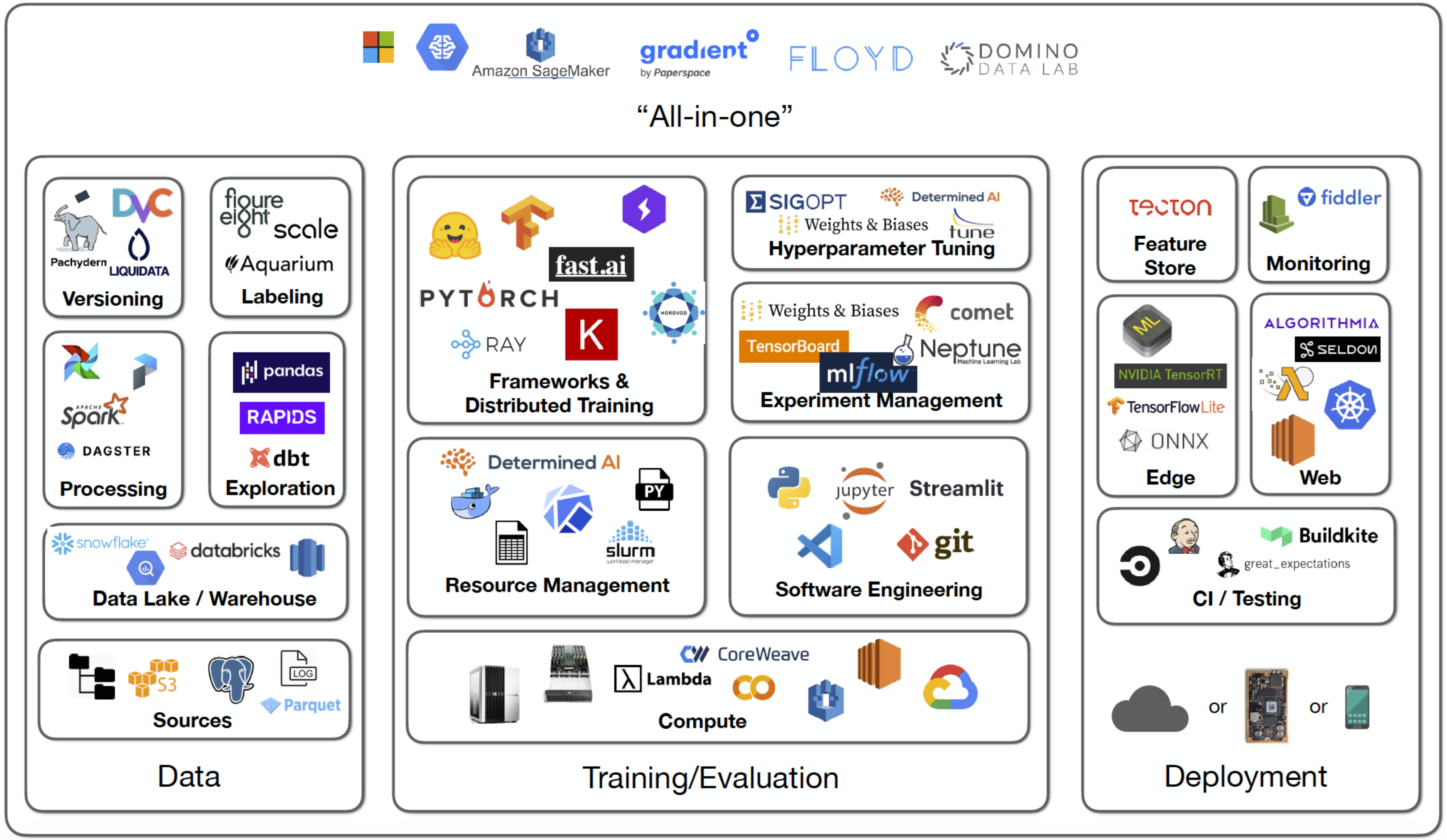
MICROSOFT

https://learn.microsoft.com/en-us/azure/architecture/ai-ml/idea/many-models-machine-learning-azure-machine-learning



MLOPS INFRASTRUCTURE & TOOLING

https://fullstackdeeplearning.com/spring2021/lecture-6/



TAKEAWAYS

- Running at huge (10k+ machines) scale requires different software stacks.
- Pretty interesting systems and design challenges.
  - try to read more papers! (e.g., BigTable, Spanner..)
- Emerging new support for ML workloads.
- Next class: **Lab Day II, Hack ZooKeeper**

REFERENCES

– [1] Malte Schwartzkopf. "What does it take to make Google work at scale?" 2015.

– [2] Jeff Dean. "Software Engineering Advice from Building Large-Scale Distributed Systems," 2007.

– [3] Jeff Dean. "Building Software Systems at Google and Lessons Learned," 2010.

– [4] Colin Scott. "Latency Numbers Every Programmer Should Know."

ACKNOWLEDGEMENT

THIS COURSE IS DEVELOPED HEAVILY BASED ON COURSE MATERIALS SHARED BY PROF. INDRANIL GUPTA, PROF. ROBERT MORRIS, PROF. MICHAEL FREEDMAN, PROF. KYLE JAMIESON, PROF. WYATT LLOYD AND PROF. ROXANA GEAMBASU. MANY APPRECIATIONS FOR GENEROUSLY SHARING THEIR MATERIALS AND TEACHING INSIGHTS.