
FINN
Cassandra

C* rocks on...

- high/fast write throughput
- schema free design (get rid of JOINs!)
- time-to-live on data
- size of data
- total load } scale out
- uptime
- tunable consistency vs availability

a few uses...

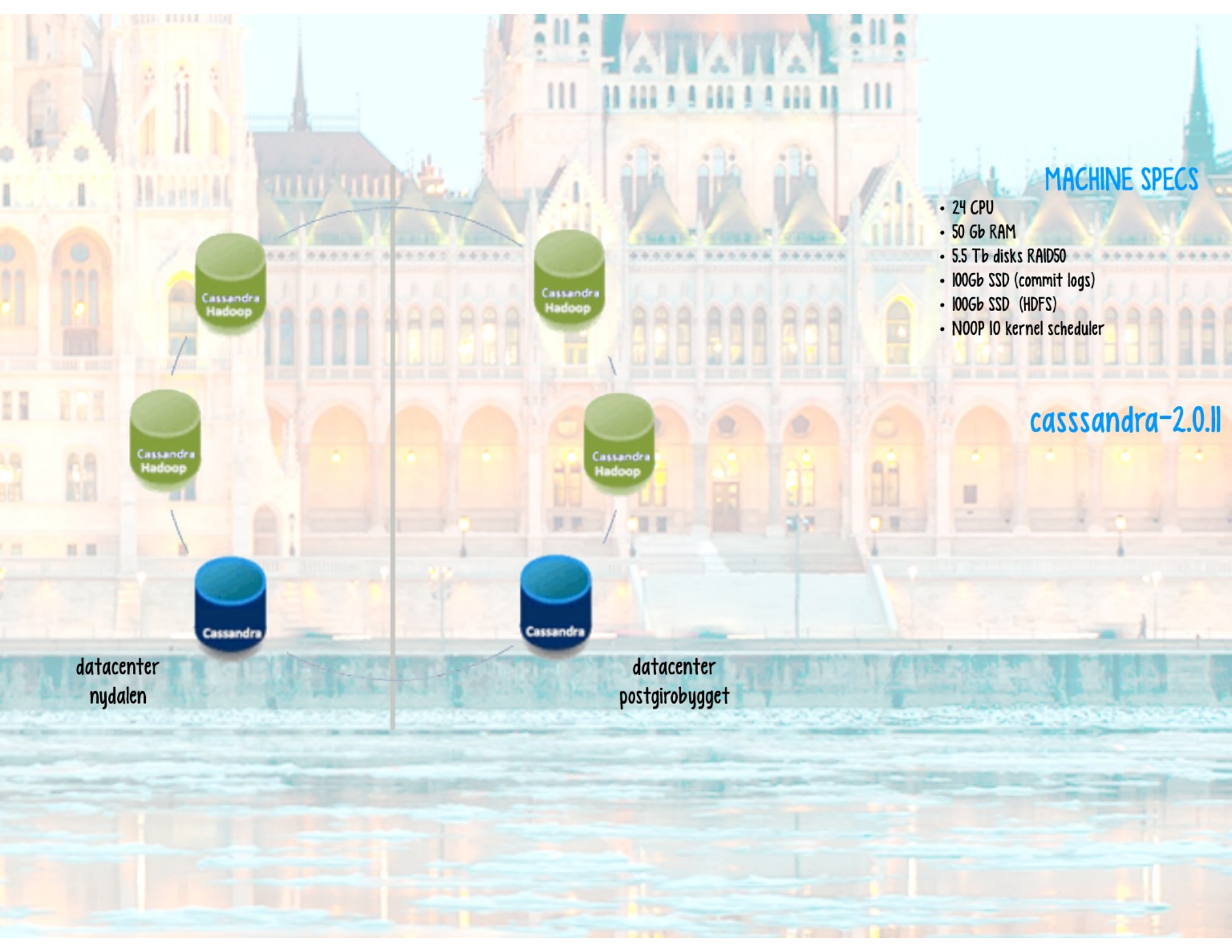
#1 Users Search History

#2 Fraud detection

- schema free design (get rid of JOINs!)
- time-to-live on data
- size of data
- total load } scale out
- uptime
- tunable consistency vs availability

a few uses...

#1 Users Search History
#2 Fraud detection
#3 IP-to-Geography
#4 Message Inbox
#5 Microservices metrics
#6 Event Statistics



MACHINE SPECS

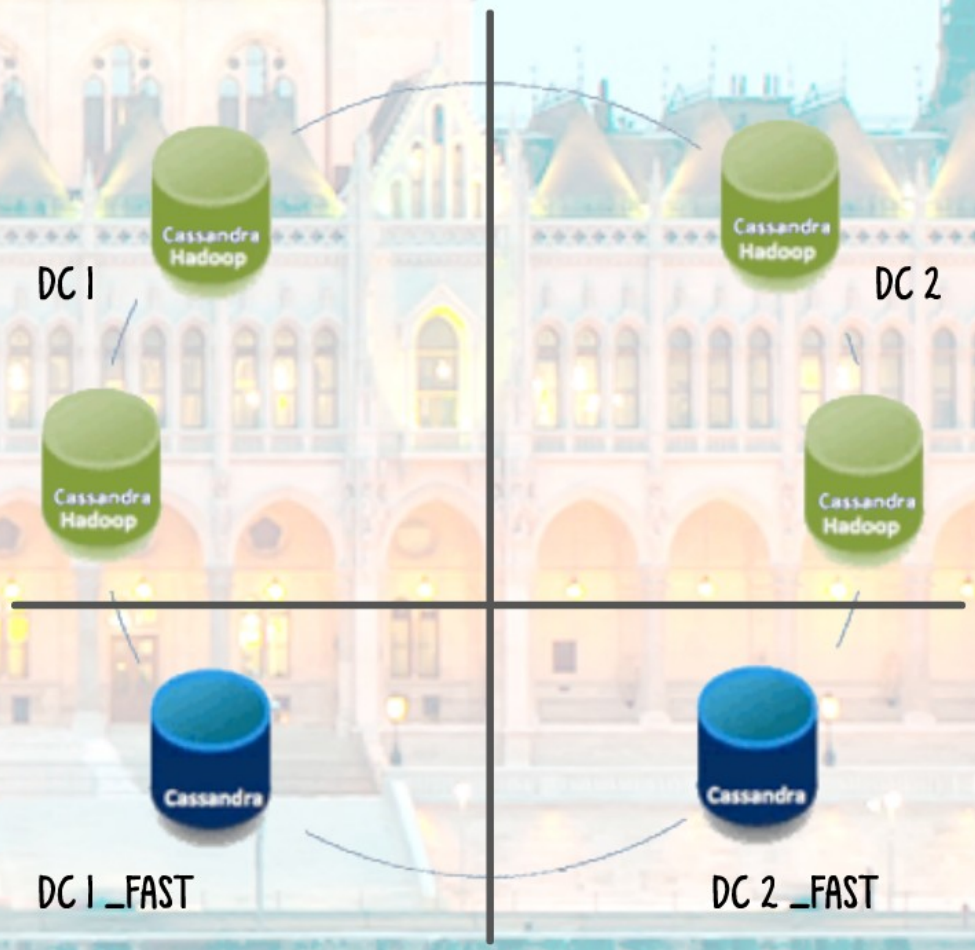
- 24 CPU
- 50 Gb RAM
- 5.5 Tb disks RAID50
- 100Gb SSD (commit logs)
- 100Gb SSD (HDFS)
- NOOP IO kernel scheduler

casssandra-2.0.11

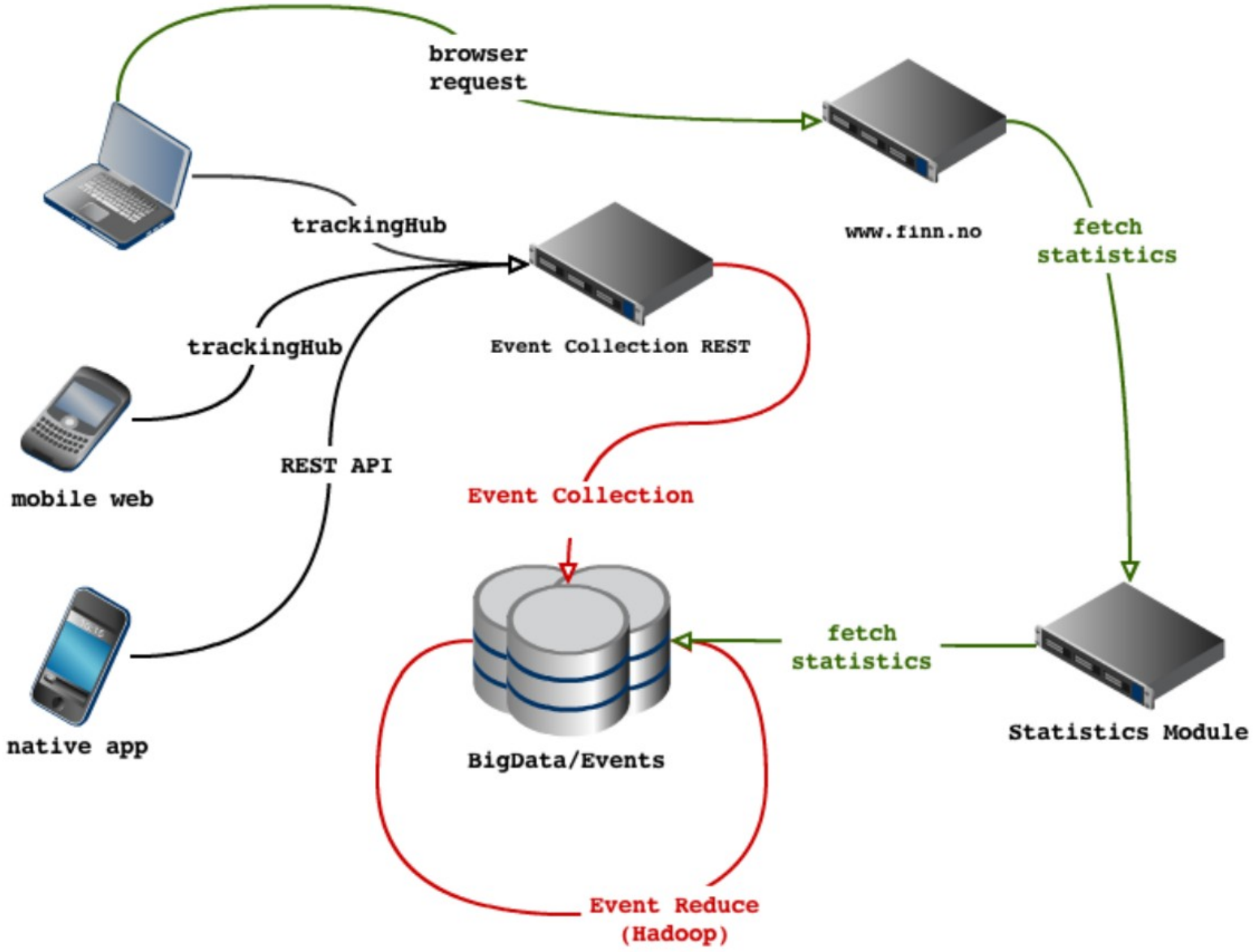
MACHINE SPECS

- 24 CPU
- 50 Gb RAM
- 5.5 Tb disks RAID50
- 100Gb SSD (commit logs)
- 100Gb SSD (HDFS)
- NOOP IO kernel scheduler

casssandra-2.0.11



Time-Series Event Tracking and Aggregation



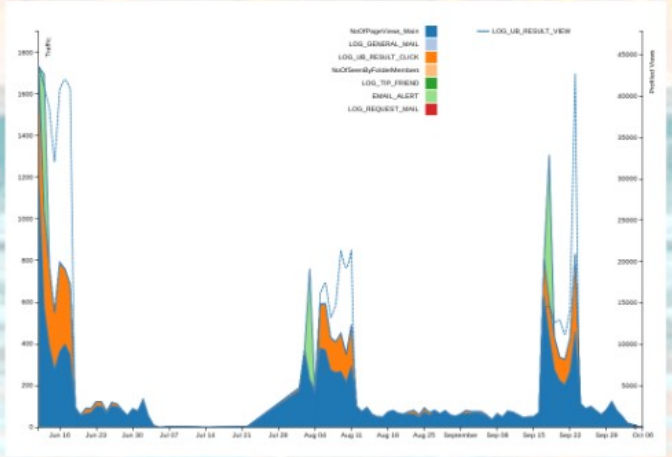
Event and **Statistics**

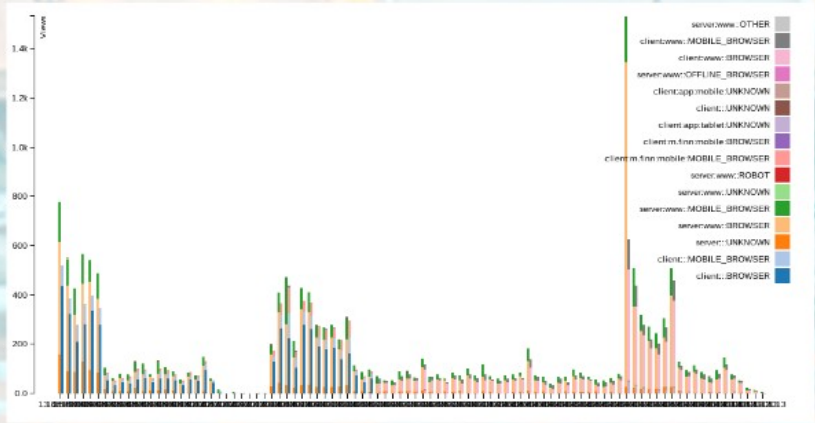
(Command Query separation)

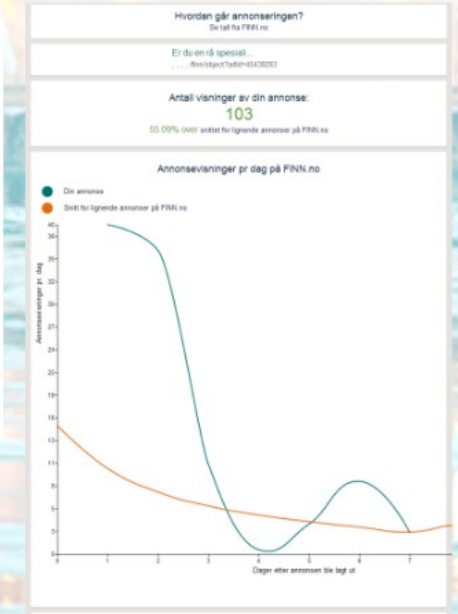
Event Reduce
(Hadoop)

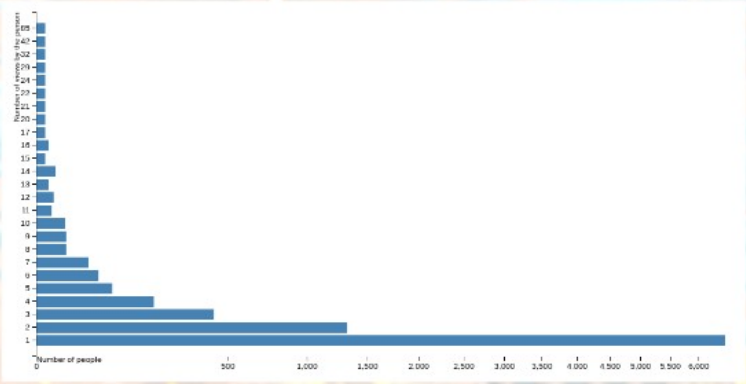
Event and **Statistics**
(Command Query separation)

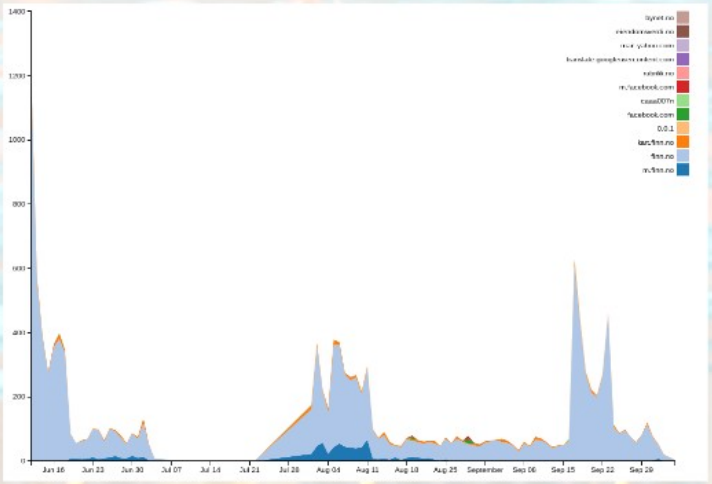
Time-Series Event Tracking and Aggregation



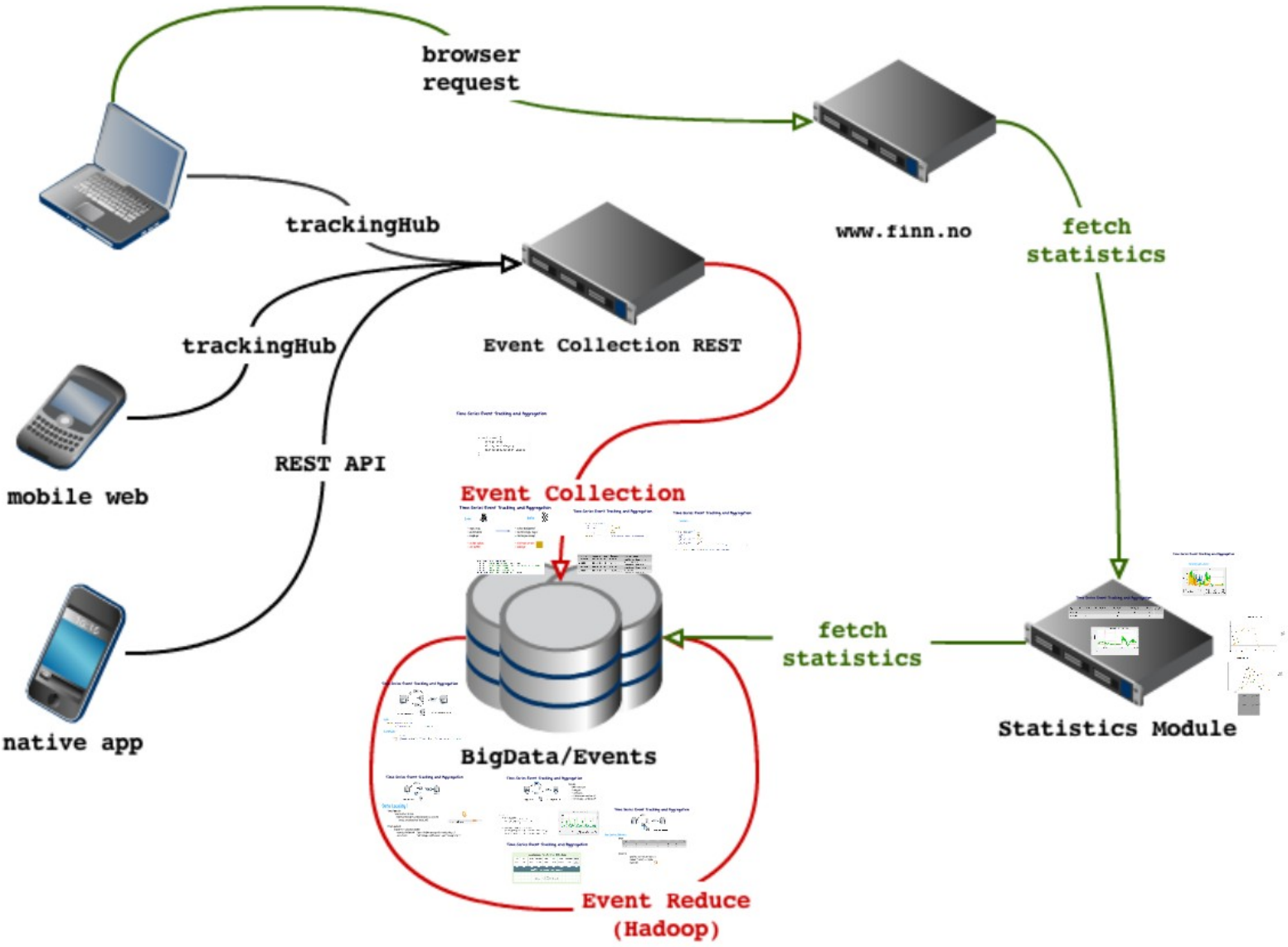








Time-Series Event Tracking and Aggregation



Event and **Statistics**

(Command Query separation)


Event Collection REST
T API
Time-Series Event Tracking and Aggregation
struct Event {
string type;
string subCategory;
map<string,string> values;
}
Event Collection
Time-Series Event Tracking and Aggregation
Scribe
Kafka
async msgs
decentralized
simple ops
archaic options
lost buffers
active development
clustered (sync msgs)
stream processing!
a lot more servers
zookeeper
Time-Series Event Tracking and Aggregation
Time-Series Event Tracking and Aggregation
fe
stati
Time-Series Event Tracking and Aggregation

Time-Series Event Tracking and Aggregation

```
struct Event {
    string type;
    string subCategory;
    map<string,string> values;
}
```

Time-Series Event Tracking and Aggregation

Scribe 

- async msgs
- decentralised
- simple ops

- archaic options
- lost buffers

Kafka 

- active development
- clustered (sync msgs)
- stream processing!

- a lot more servers
- zookeeper

```
Properties props = new Properties();
props.put("metadata.broker.list", BROKER_LIST);
props.put("key.serializer.class", "kafka.serializer.StringEncoder");
props.put("producer.type", "sync");
props.put("request.required.acks", "-1");
return new Producer<>(new ProducerConfig(props));
```

Time-Series Event Tracking and Aggregation

```
CREATE TABLE events (
  timebucket                text,
  timestamp                 timeuuid,
  type                      text,
  subcategory               text,
  keys_and_values           text,
  PRIMARY KEY               (timebucket, type, timestamp)
);
```

timebucket_row	timestamp	type	subcategory	keys_and_values
1369164000	1369164000	AD	PageView	{"ad.user":"711130430", "user.uniqueVisitorId":"1799736047",...}
1369164000	1369164001	AD	PageView	{"ad.user":"711130430", "user.uniqueVisitorId":"1799736047",...}
1369164000	1369164002	AD	EmailSent	{"ad.user":"711130430", "user.uniqueVisitorId":"1799736047",...}
1369164000	1369164003	AD	MarkedFavourite	{"ad.user":"711130430", "user.uniqueVisitorId":"1799736047",...}

Time-Series Event Tracking and Aggregation

the real world...

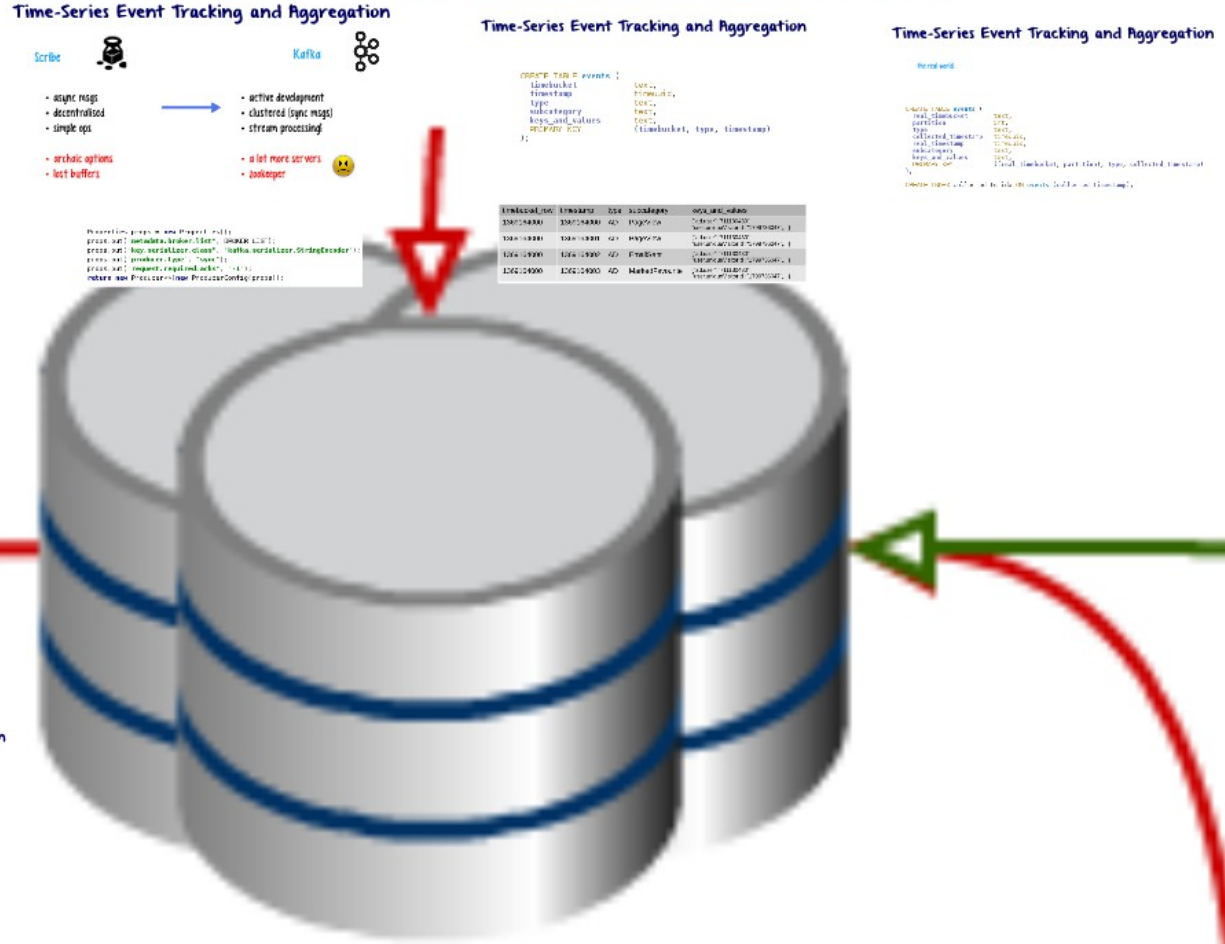
```
CREATE TABLE events (
  real_timebucket       text,
  partition             int,
  type                  text,
  collected_timestamp   timeuuid,
  real_timestamp        timeuuid,
  subcategory           text,
  keys_and_values       text,
  PRIMARY KEY           ((real_timebucket, partition), type, collected_timestamp)
);

CREATE INDEX collected_ts_idx ON events (collected_timestamp);
```

Time-Series Event Tracking and Aggregation

```
CREATE TABLE events (
  timebucket                text,
  timestamp                 timeuuid,
  type                      text,
  subcategory               text,
  keys_and_values           text,
  PRIMARY KEY               (timebucket, type, timestamp)
);
```

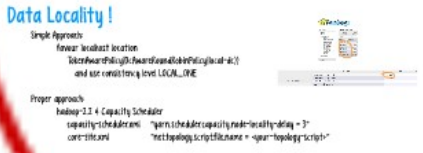
timebucket_row	timestamp	type	subcategory	keys_and_values
1369164000	1369164000	AD	PageView	{"ad.user":"711130430", "user.uniqueVisitorId":"1799736047",...}
1369164000	1369164001	AD	PageView	{"ad.user":"711130430", "user.uniqueVisitorId":"1799736047",...}
1369164000	1369164002	AD	EmailSent	{"ad.user":"711130430", "user.uniqueVisitorId":"1799736047",...}
1369164000	1369164003	AD	MarkedFavourite	{"ad.user":"711130430", "user.uniqueVisitorId":"1799736047",...}

Event Collection



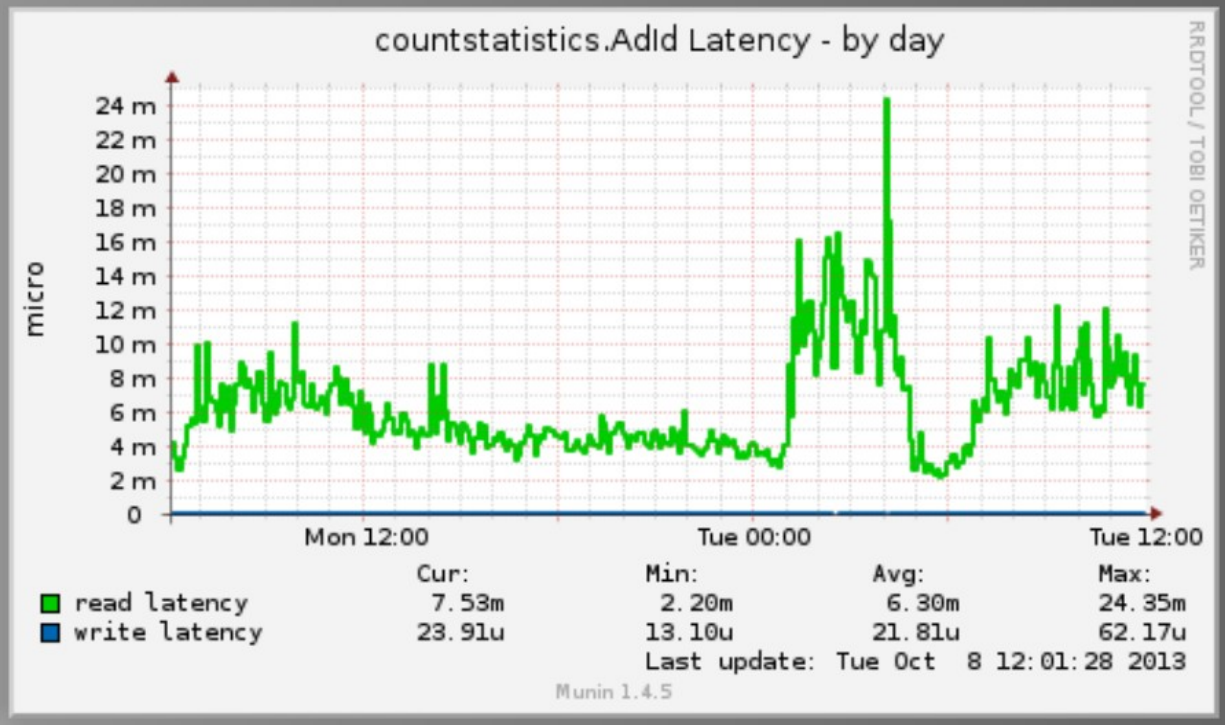
fete
statis

BigData/Events

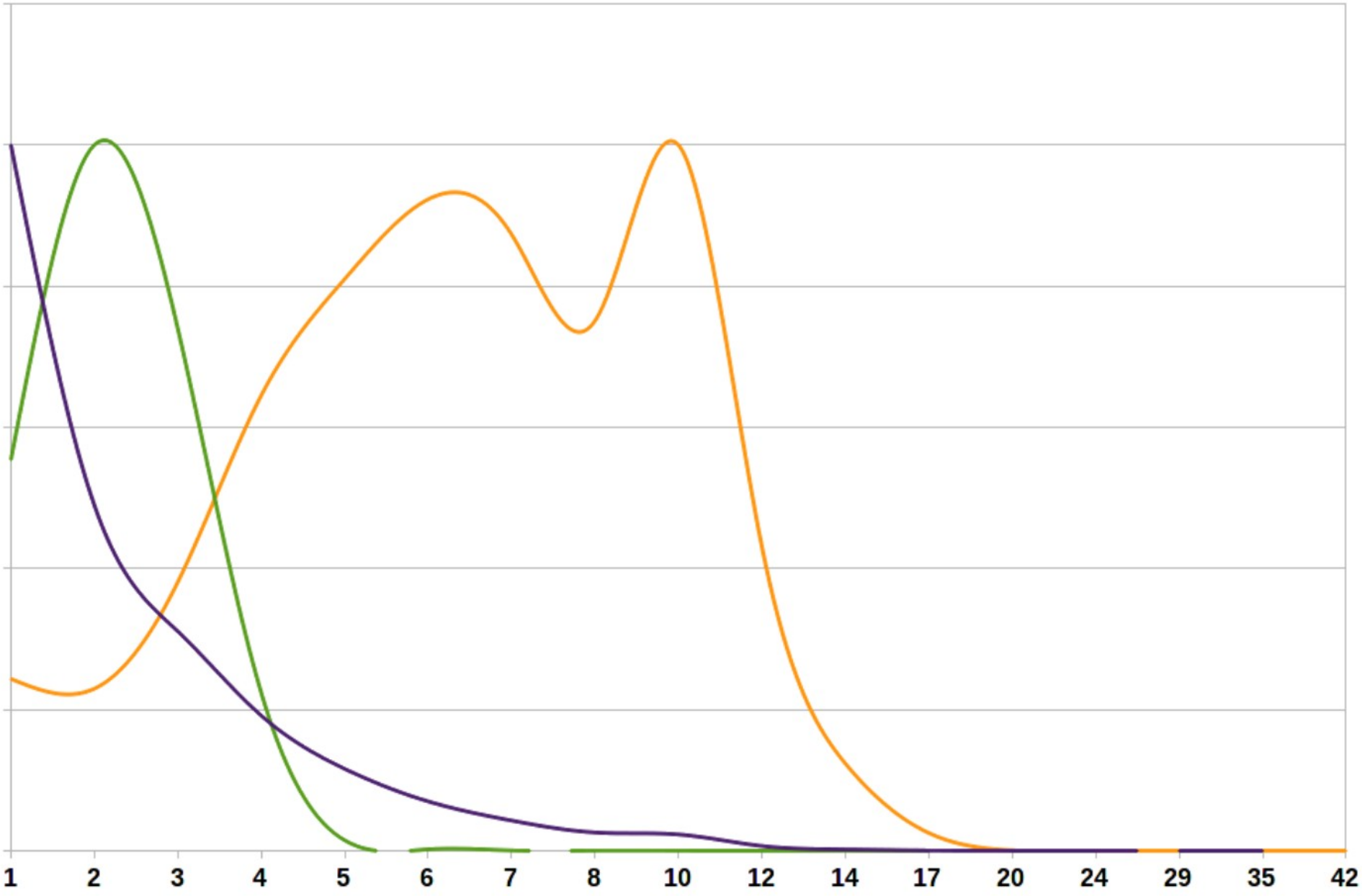


Time-Series Event Tracking and Aggregation

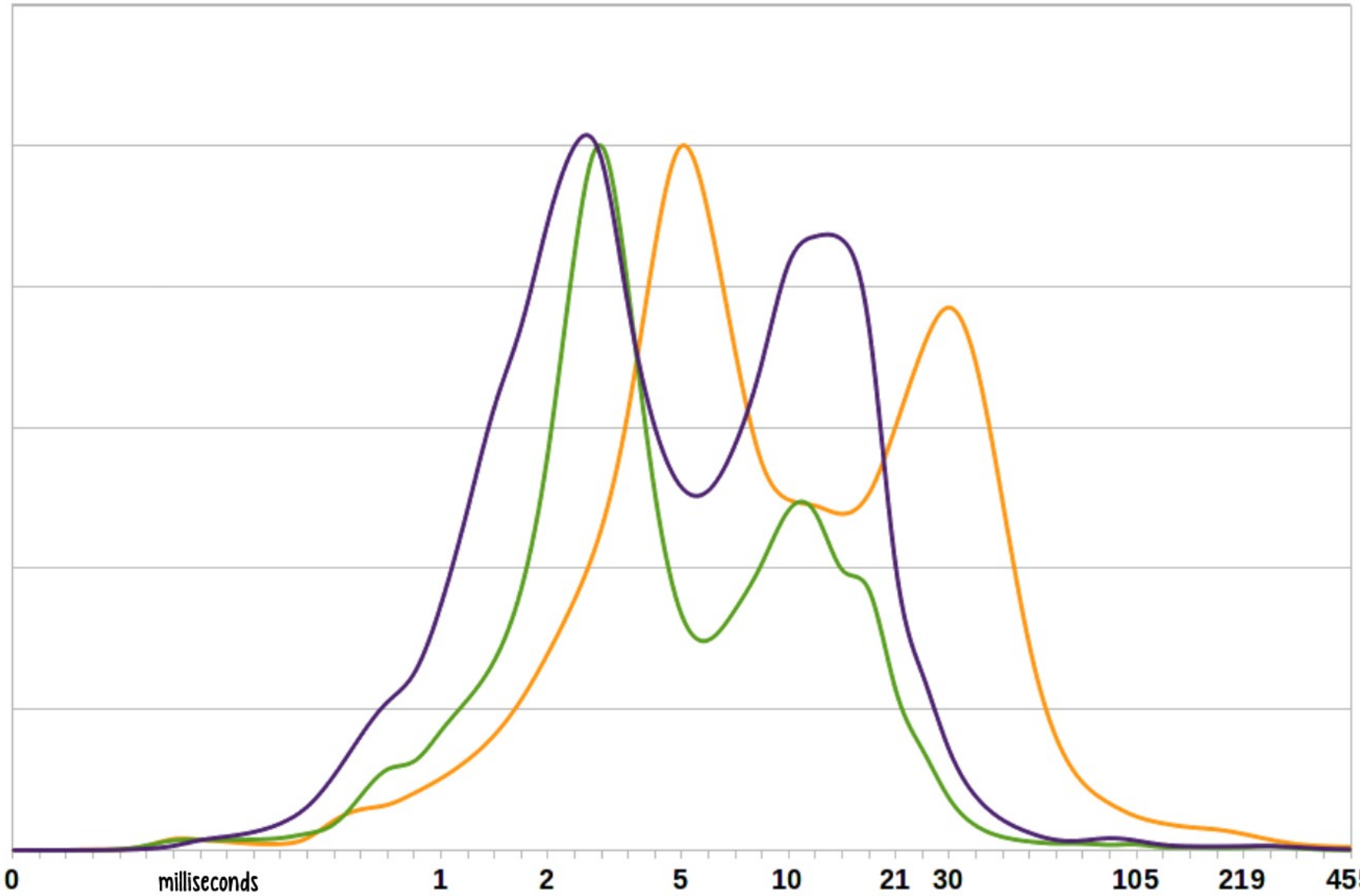
d	HOUR_201328111: PageView	HOUR_201328112: EmailSent	DAY_2012338: PageView	DAY_2012339: EmailSent	TOTAL: PageView	TO Emai
119	10	2	20	4	50	
052	23	3	102	4	234	



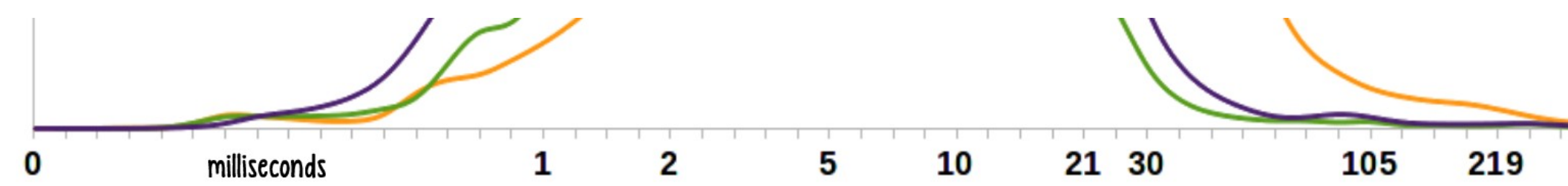
SSTables per Read

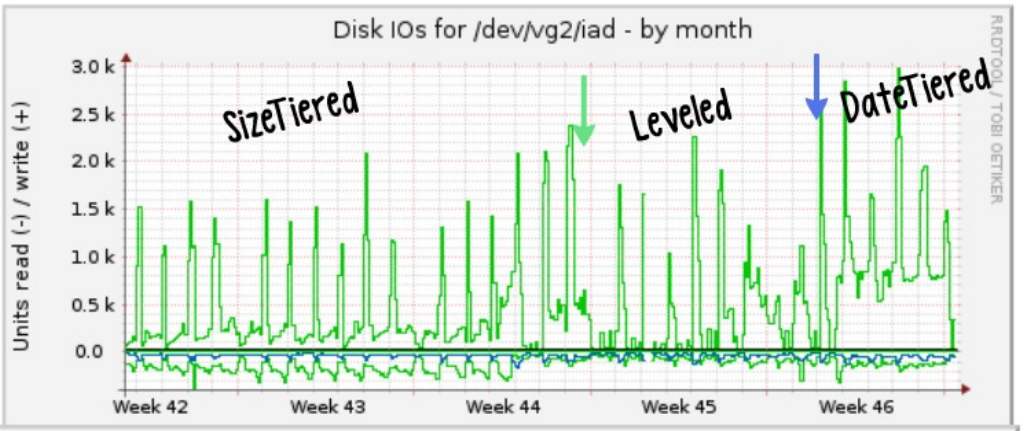


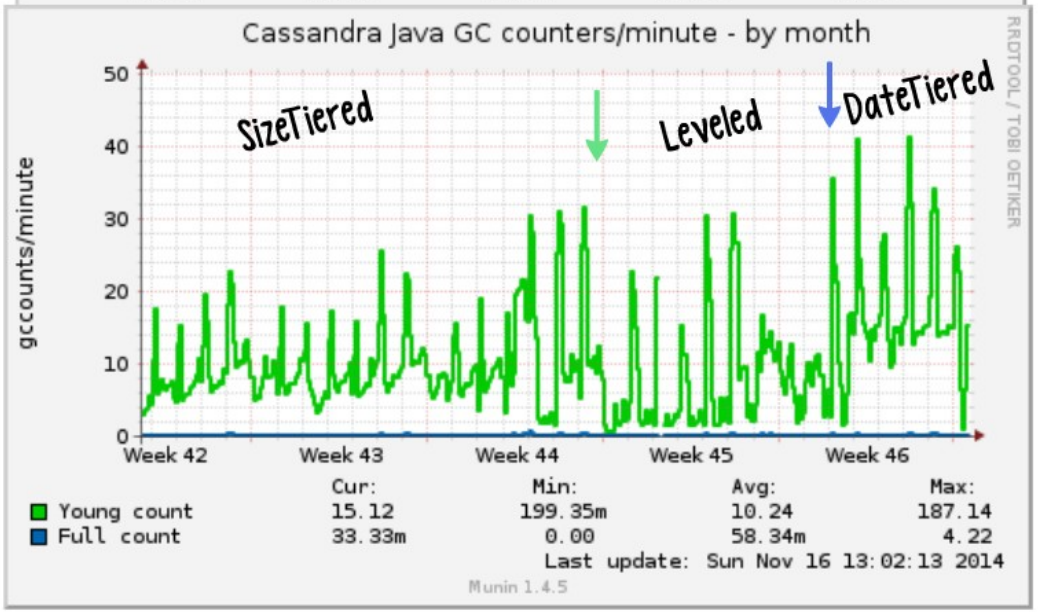
Read Latency



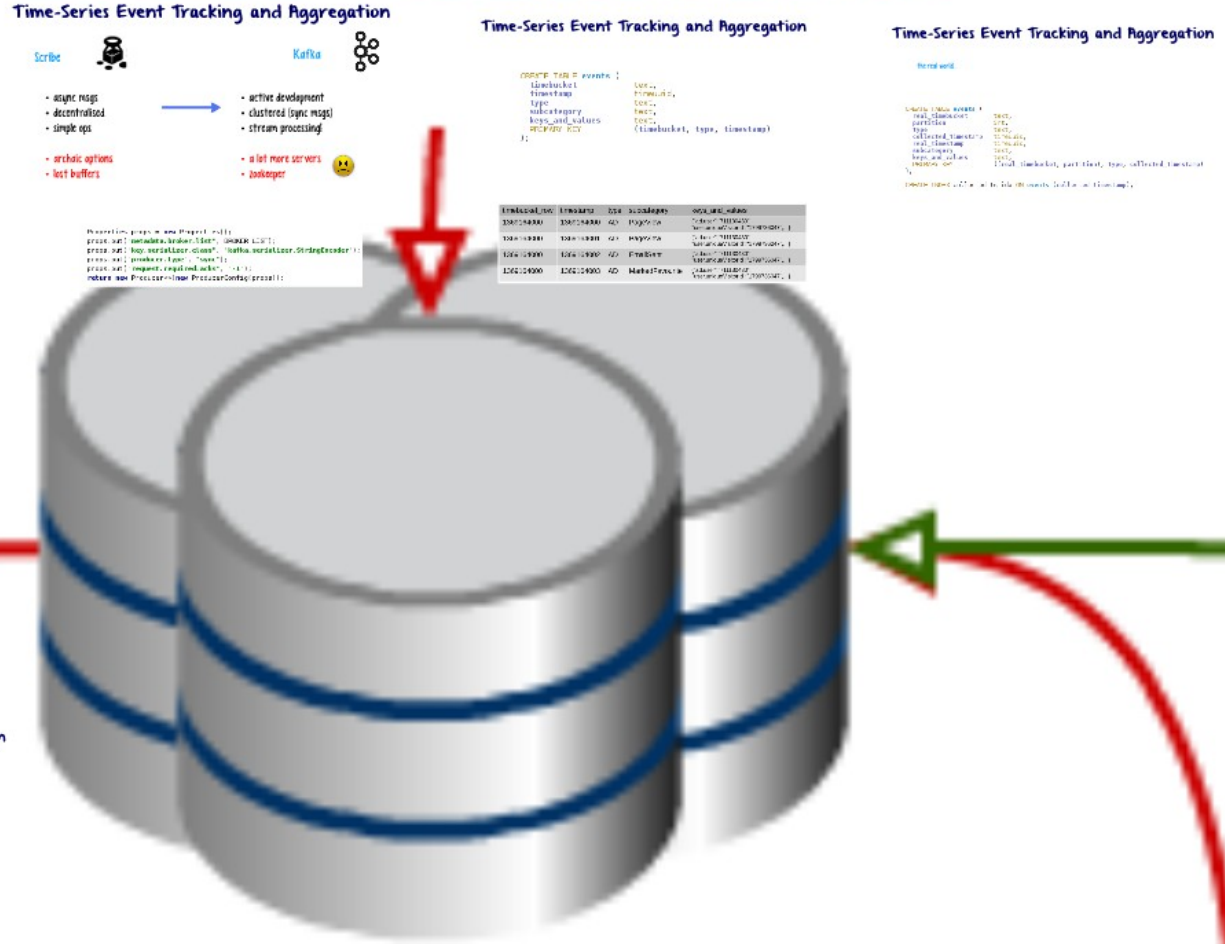
Disk IOs for /dev/vg2/iad - by month


0
milliseconds
1
2
5
10
21
30
105
219


Disk IOs for /dev/vg2/iad - by month
Units read (-) / write (+)
3.0 k
2.5 k
2.0 k
1.5 k
1.0 k
0.5 k
0.0
SizeTiered
Leveled
DateTiered
Week 42
Week 43
Week 44
Week 45
Week 46
RRDTOOL / TOBI OETIKER

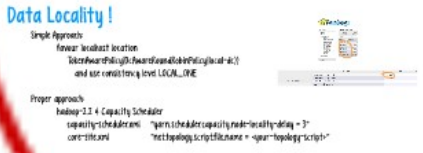

Cassandra Java GC counters/minute - by month
gccounts/minute
50
40
30
20
10
0
SizeTiered
Leveled
DateTiered
Week 42
Week 43
Week 44
Week 45
Week 46
RRDTOOL / TOBI OETIKER
Cur:
Min:
Avg:
Max:
Young count
15.12
199.35m
10.24
187.14
Full count
33.33m
0.00
58.34m
4.22
Last update: Sun Nov 16 13:02:13 2014
Munin 1.4.5

Event Collection

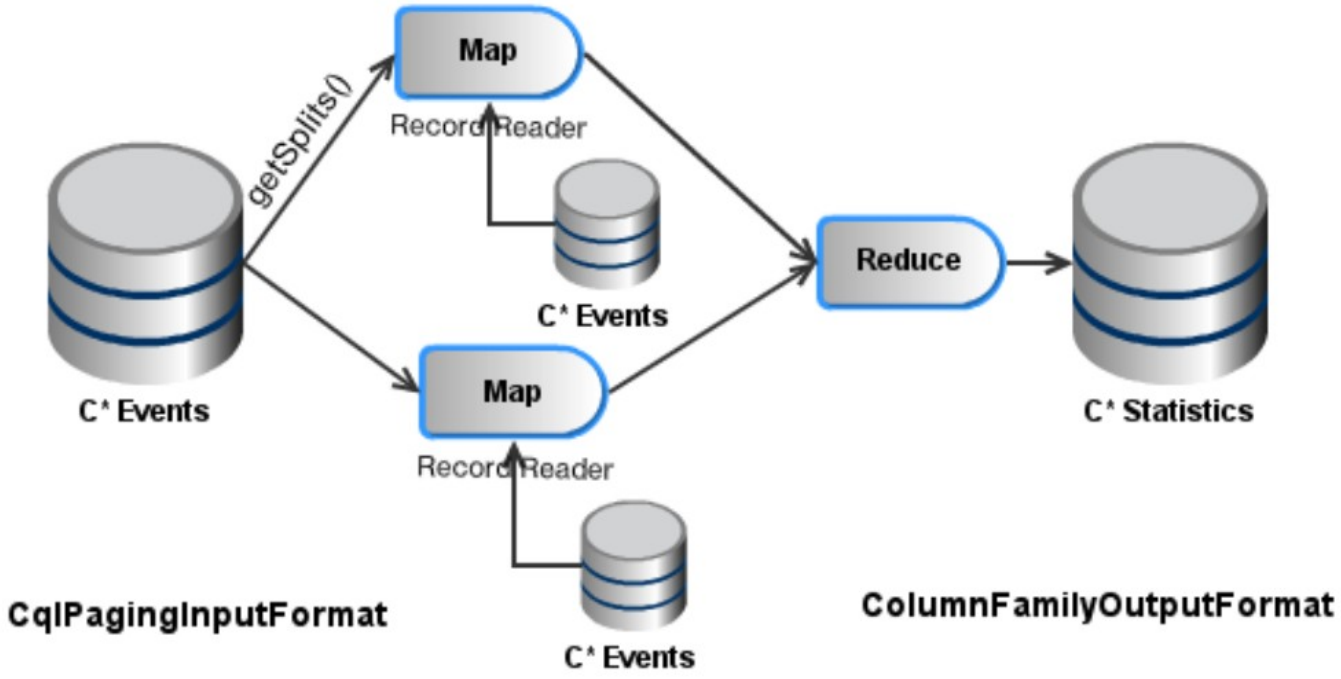


fete
statis

BigData/Events



ne-Series Event Tracking and Aggregation



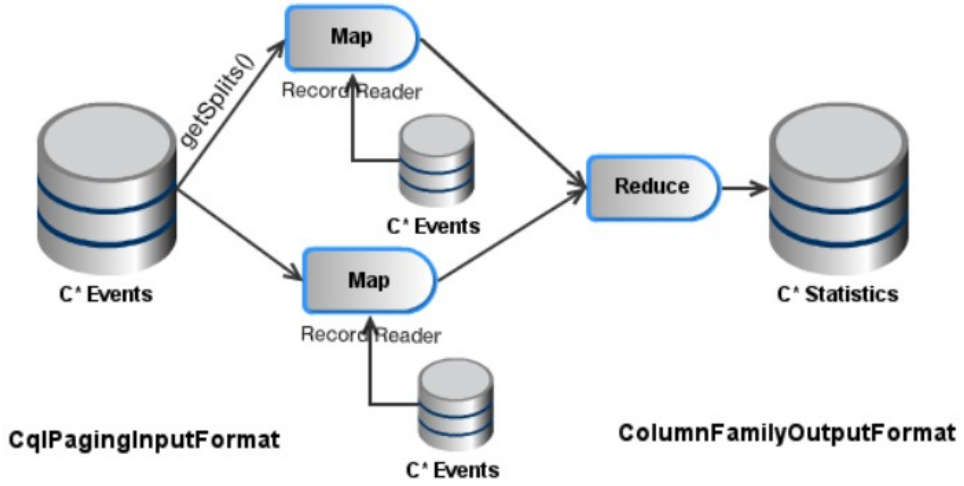
lits

```
SELECT timebucket FROM events
    WHERE type = 'AD'
        AND timebucket IN ('startMinute..endMinute')
```

cordReader

```
SELECT * FROM events
    WHERE timebucket = 'currentMinute' AND type = 'AD'
        AND timestamp >= minTimeuuid('start') AND timestamp < maxTimeuuid('endTime')
    LIMIT 100
```

Time-Series Event Tracking and Aggregation



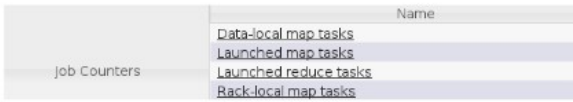
ata Locality !

Simple Approach:

favour localhost location

TokenAwarePolicy(DcAwareRoundRobinPolicy(local-dc))

and use consistency level LOCAL_ONE



Proper approach:

hadoop-2.2 & Capacity Scheduler

capacity-scheduler.xml "yarn.scheduler.capacity.node-locality-delay = 3"

core-site.xml "net.topology.script.file.name = <your-topology-script>"

- Cluster
  - About
  - Nodes
  - Applications
    - NEW
    - NEW_SAVING
    - SUBMITTED
    - ACCEPTED
    - RUNNING
    - REMOVING
    - FINISHING
    - FINISHED
    - FAILED
    - KILLED
  - Scheduler
- Tools

Cluster Metrics

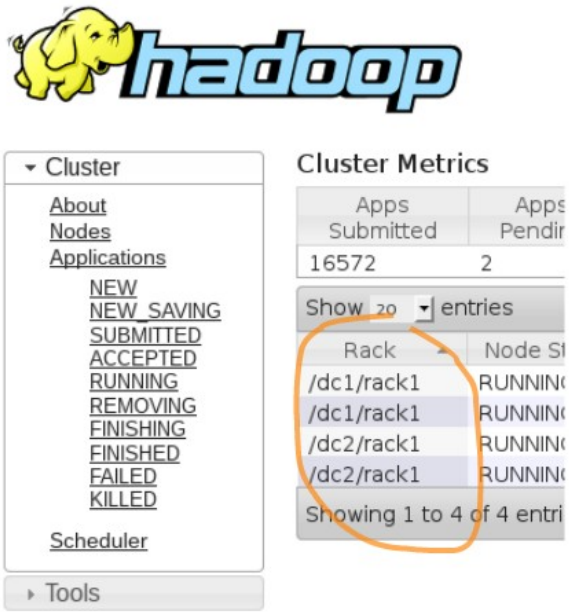
Apps Submitted	Apps Pendir
16572	2

Show 20 entries

Rack	Node St
/dc1/rack1	RUNNIN
/dc1/rack1	RUNNIN
/dc2/rack1	RUNNIN
/dc2/rack1	RUNNIN

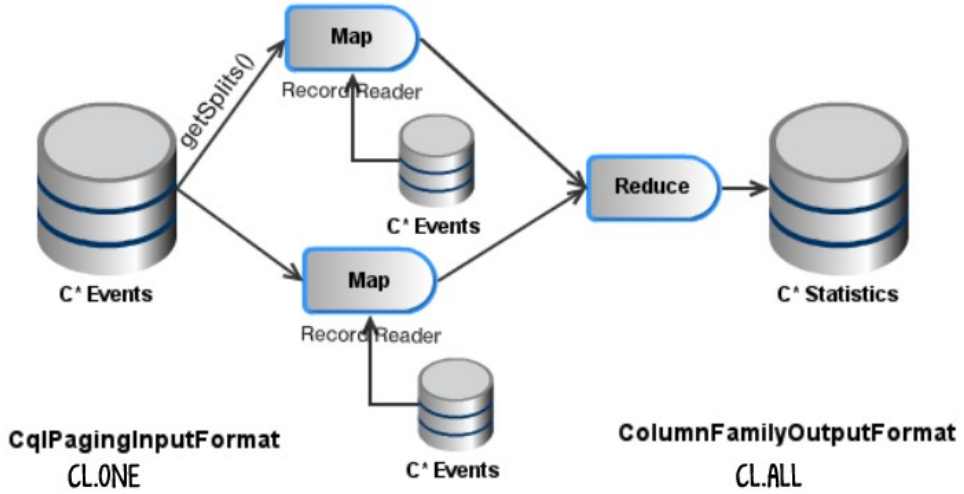
Showing 1 to 4 of 4 entri

Name	
Data-local map tasks	1513



	Name	Total
Job Counters	Data-local map tasks	15139
	Launched map tasks	15140
	Launched reduce tasks	1
	Rack-local map tasks	1

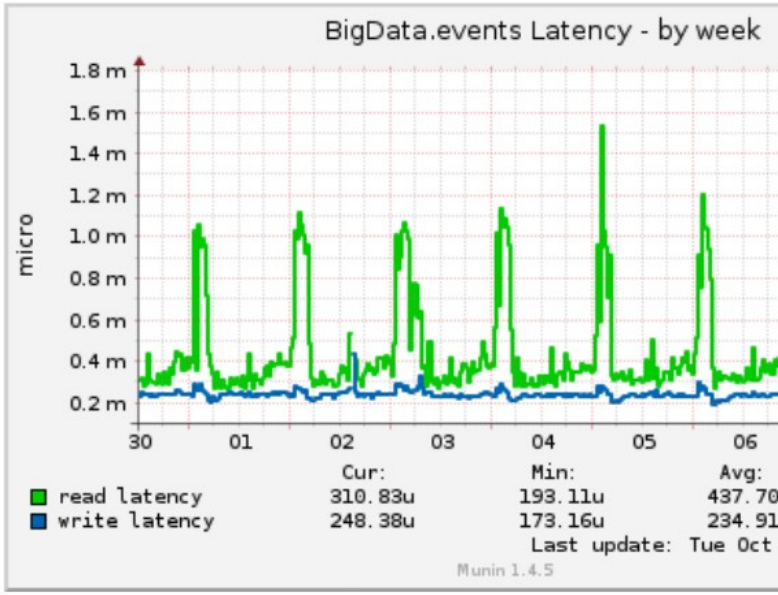
Time-Series Event Tracking and Aggregation



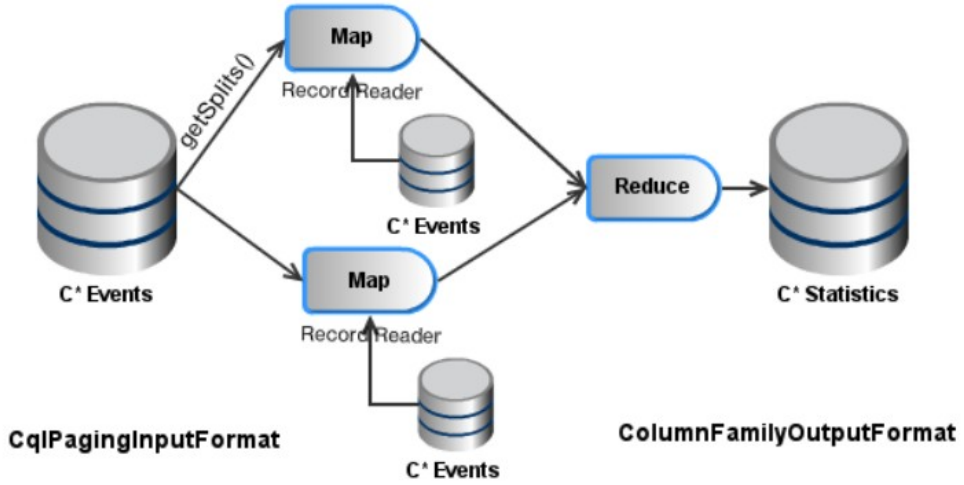
Each day:

- 5000+ minute jobs
- 7 daily jobs
- + ad-hoc jobs
- ~ 1 billion records read
- ~ 150M records writte

```
s):
vent in events:
d = event.getValues()["ad.id"]
ubcategory = (event.subcategory, 1)
mit(id, subcategory)

, subcategories):
ubcategory, count in subcategories:
ncrementCassandraCounter(id, subcategory, count, HOUR_XX)
ncrementCassandraCounter(id, subcategory, count, DAY_ZZZ)
ncrementCassandraCounter(id, subcategory, count, TOTAL)
```



Time-Series Event Tracking and Aggregation



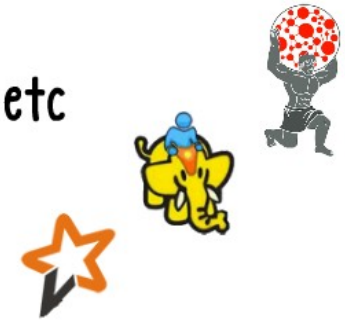
tric Statistics

Simple...

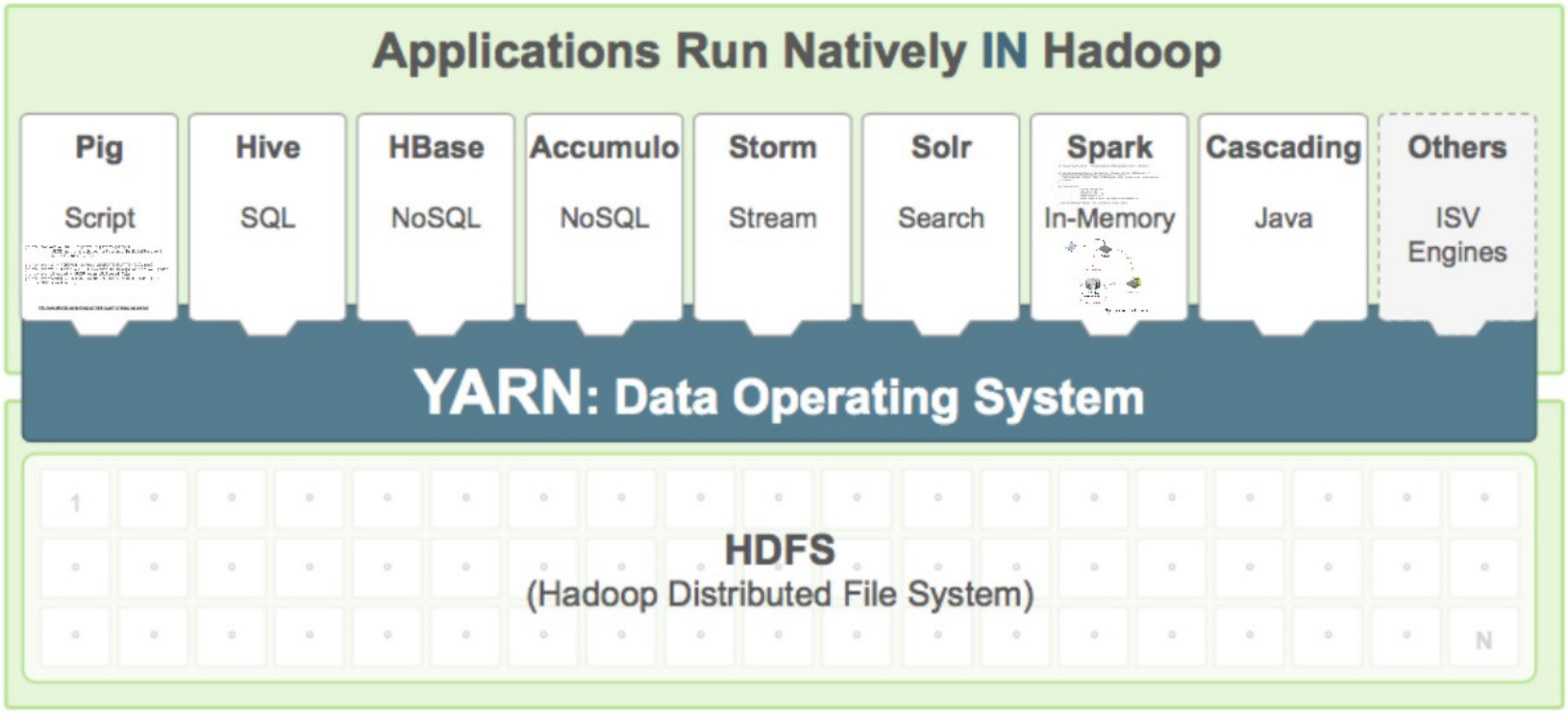
UserId	TOTAL:Ski	TOTAL:Java	TOTAL:Tesla	TOTAL:Bicycle	TOTAL:Oslo	TOTAL:Bergen
706119	10	2	20	4	50	5
706052	23	3	102	4	234	10

Advanced...

- graphing: user->ad, ad->user, etc
- Mahout ("Taste") --> Myrrix
- Spark ALS



Time-Series Event Tracking and Aggregation

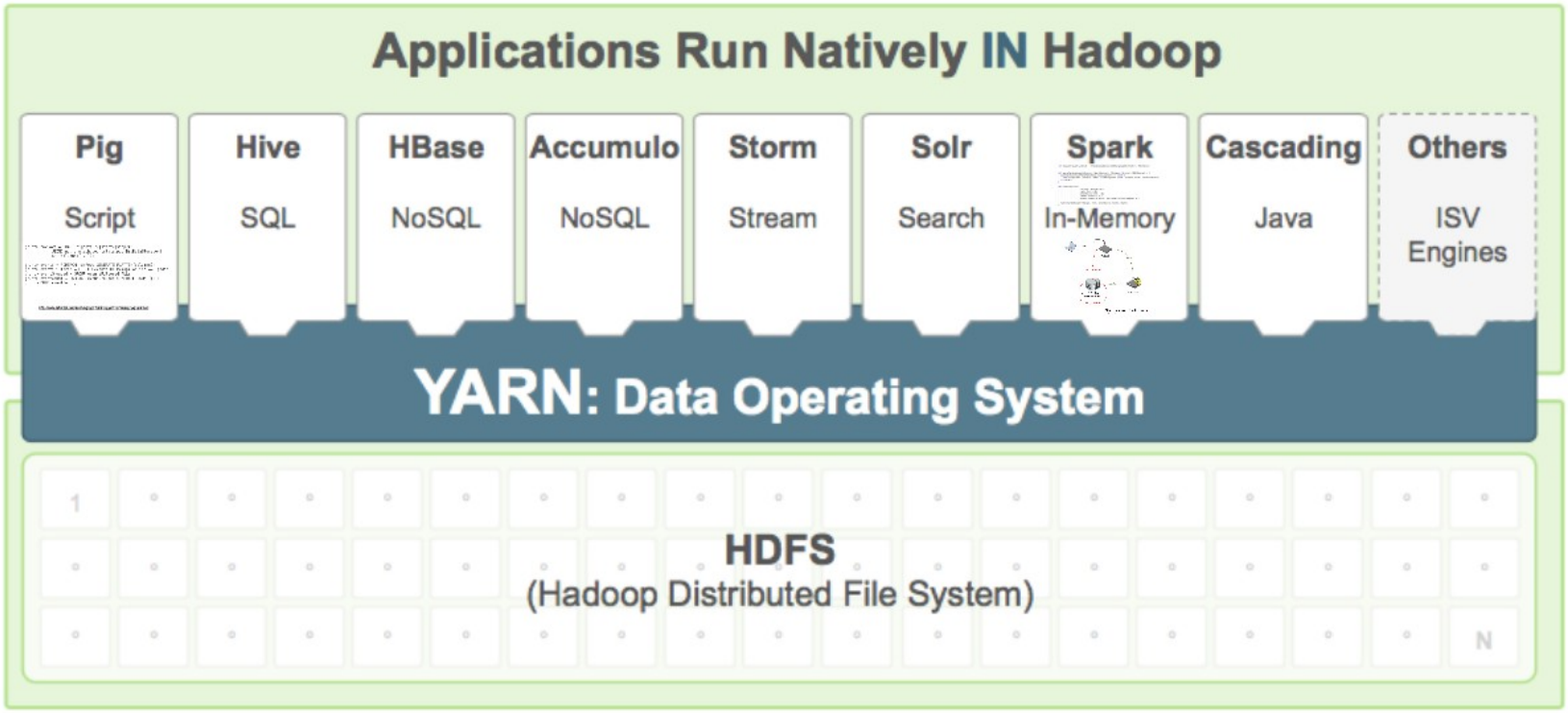


Script

```
grunt> rawRows = LOAD 'bigdata://$start/$stop/AD'
           USING no.finntech.countstats.pig.BigDataStorage()
           AS (columns:{...});

grunt> events = FOREACH rawRows GENERATE FLATTEN(columns);
grunt> eventsFiltered = FILTER events BY kvmap#'ad.id' == '$adid';
grunt> eventGrouped = GROUP eventsFiltered ALL;
grunt> eventCount = FOREACH eventGrouped GENERATE COUNT($1);
grunt> DUMP eventCount;
```

http://www.datastax.com/dev/blog/cql3-table-support-in-hadoop-pig-and-hive

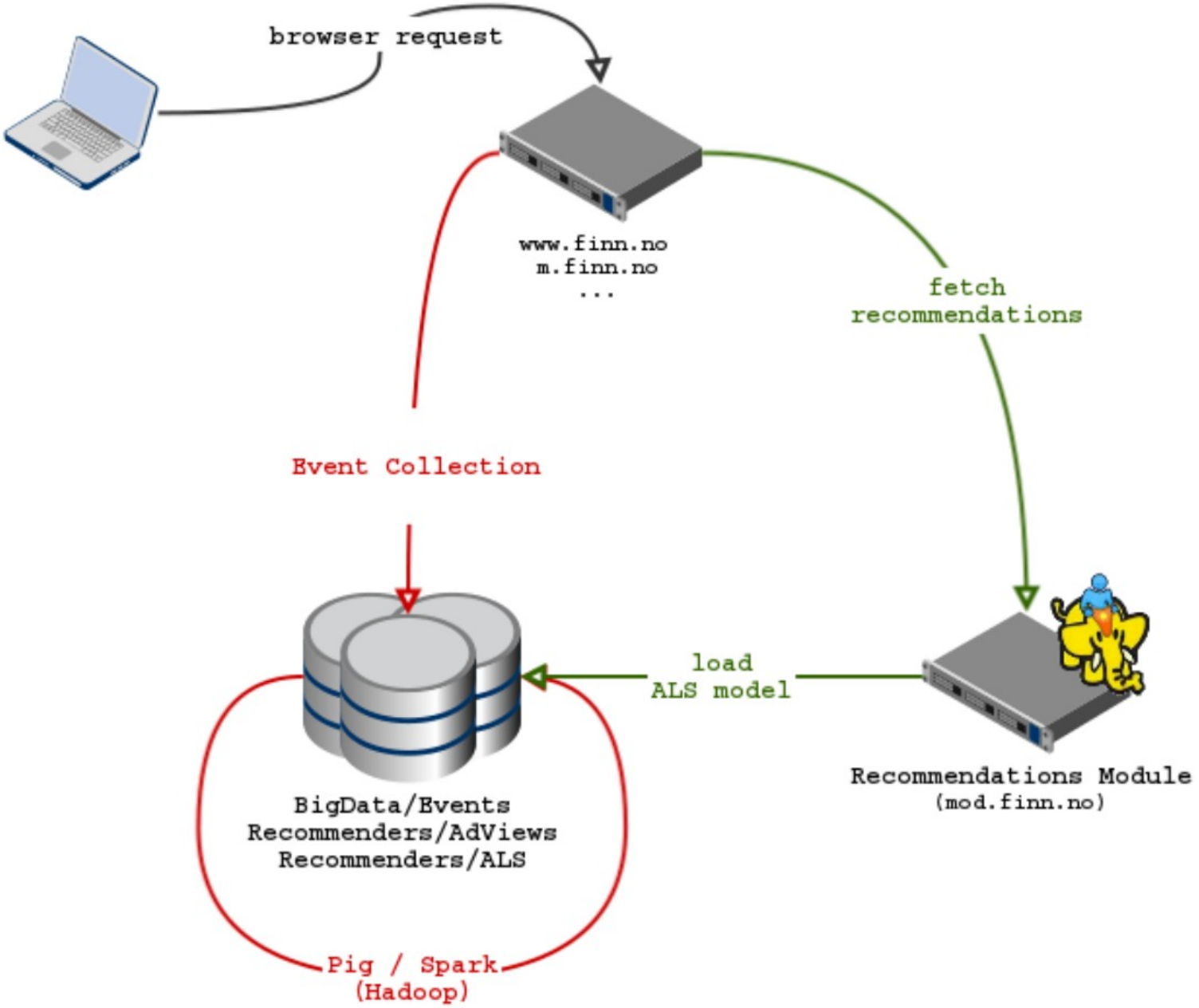
Time-Series Event Tracking and Aggregation



Spark

```scala
val als_matrix_for_mahout = trainImplicit(parseRatings(getContext(), fileName))


def parseRatings(sparkContext: SparkContext, fileName: String): RDD[Rating] = {
  sparkContext.textFile(fileName).map(_.split(",") match {
    case Array(user, product, rate) => Rating(user.toInt, product.toInt, rate.toDouble)
  }).cache()
}

def trainImplicit(
                  ratings: RDD[Rating],
                  rank: Int = 40,
                  iterations: Int = 20,
                  lamda: Double = 0.1,
                  alpha: Double = 60.0): MatrixFactorizationModel = {

  ALS.trainImplicit(ratings, rank, iterations, lamda, alpha)
}
```

In-Memory



Recommendations

(Command Query separation)

C* is moving way quick!

- Secondary Indexes
- Compression
- CQL3 (+MapReduce +Spark)
- CQL tracing
- Automatic pagination
- async and unlogged batch statements
- fluent api to cql java driver
- Counters-2 !! (Cassandra-2.1)

lessons learnt

- During C* ops
  - plan carefully, test strategies, test clients,
  - easy to bump CL.ONE -> CL.QUORUM,
  - stop Hadoop
- Heed disk latency + utilisation
  - >20% asking for trouble
  - SSD for commit-log, separate SSD for HDFS,
    (Cassandra writes sequential)
- Comprehensive monitoring - detect problems quick
  - C* is robust and will hide problems
  - monitor gc, and pending actions growing,
  - look out for spikes in 95th percentile
- Stay under capacity!
  - easy backup, repair, streaming, etc
- Avoid OrderPreservingPartitioner
- Avoid custom serialised data (transparency is gold)
- Avoid skinny rows on non-commodity machines
- CQL3 rocks (always!)
  - Use json over maps (until you need maps)
- Don't run JobTracker+NameNode on a C* node
- Upgrading to vnodes tedious (~2 months)