Ron Bodkin
Founder & CEO, Think Big

June 2013

Real-time Big Data Analytics with Storm

Leading Provider of
Data Science and Engineering Services

Accelerating Your Time to Value

IMAGINE
Strategy
and Roadmap

ILLUMINATE
Training
and Education

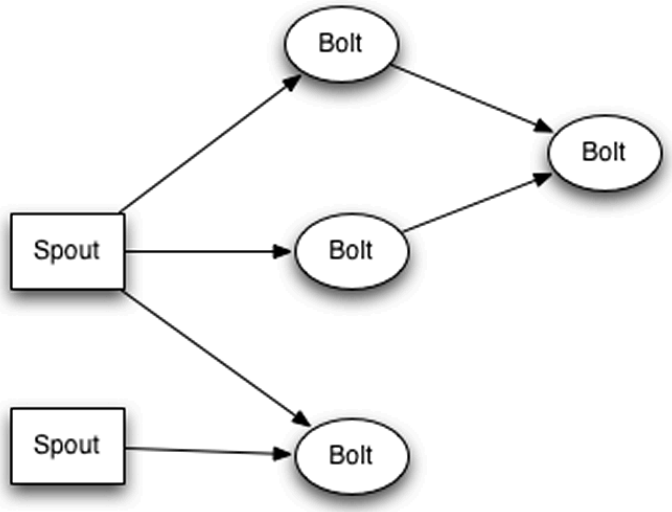
IMPLEMENT
Hands-On
Data Science and
Data Engineering

Agenda

- What is Storm?
- When is Storm appropriate?
- API Options
- Use Cases

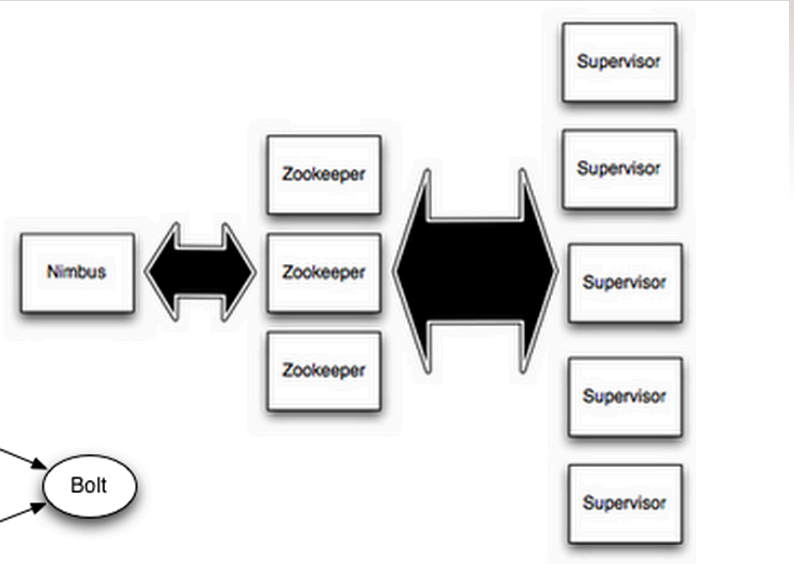
Storm Overview

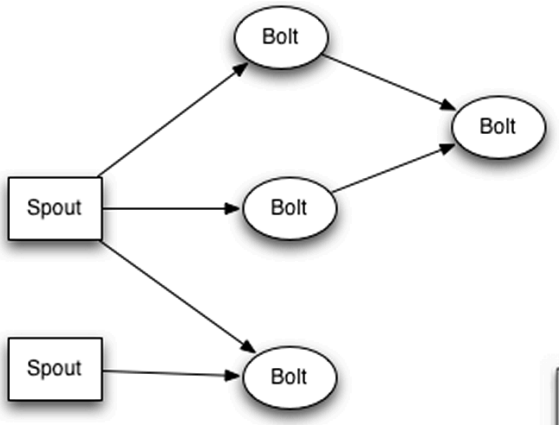
- DAG Processing of never ending streams of data
  - Open Source: https://github.com/nathanmarz/storm/wiki
  - Used at Twitter plus > 24 other companies
  - Reliable - At Least Once semantics
  - Think MapReduce for data streams
  - Java / Clojure based
  - Bolts in Java and 'Shell Bolts'
  - Not a queue, but usually reads from a queue.
- Related:
  - S4, CEP
- Compromises
  - Static topologies & cluster sizing – Avoid messy dynamic rebalancing
  - Nimbus – SPOF
- Strong Community Support, emerging commercial support

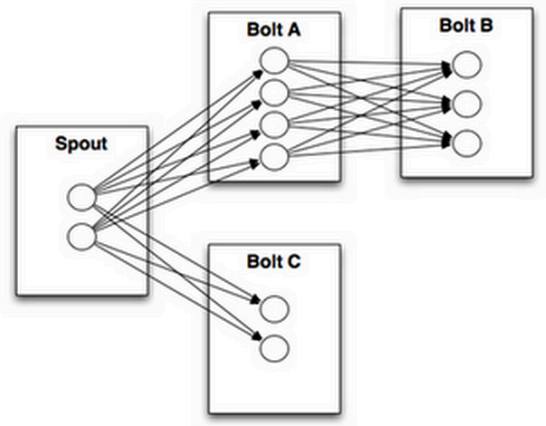


Storm Concepts

- Cluster
  - Nimbus
  - Zookeeper
  - Supervisor, Workers
- Topology
- Streams
- Tuple
- Spout
- Bolt
- Stream Groupings
  - Shuffle, Field
- Trident
- DRPC







What is Real-Time?

- Low latency
  - Query response
  - Data refresh
  - End-to-end response
- … nanoseconds, milliseconds, seconds, or minutes depending on your problem

Why Real-time?

- Better end-user experience
  - Ex: View an ad, see the counter move.
- Operational intelligence
  - Low latency analysis
  - Operational intelligence
- Event response
  - Content half life measured at 3 hours (H Mason: http://bit.ly/nu7IDw)
  - Path to additional real-time capabilities
  - Scalable analysis
  - Example: Trend analysis to recommend ‘hot’ articles.

Why Storm?

- Scalable way to manage queue readers and logic pipeline
- Much better than roll your own
- Reliable (Message guarantees, fault tolerant)
- Multi-node scaling (1MM messages / 10 nodes)
- It works
- Open source community
- See also: https://github.com/nathanmarz/storm/wiki/Rationale

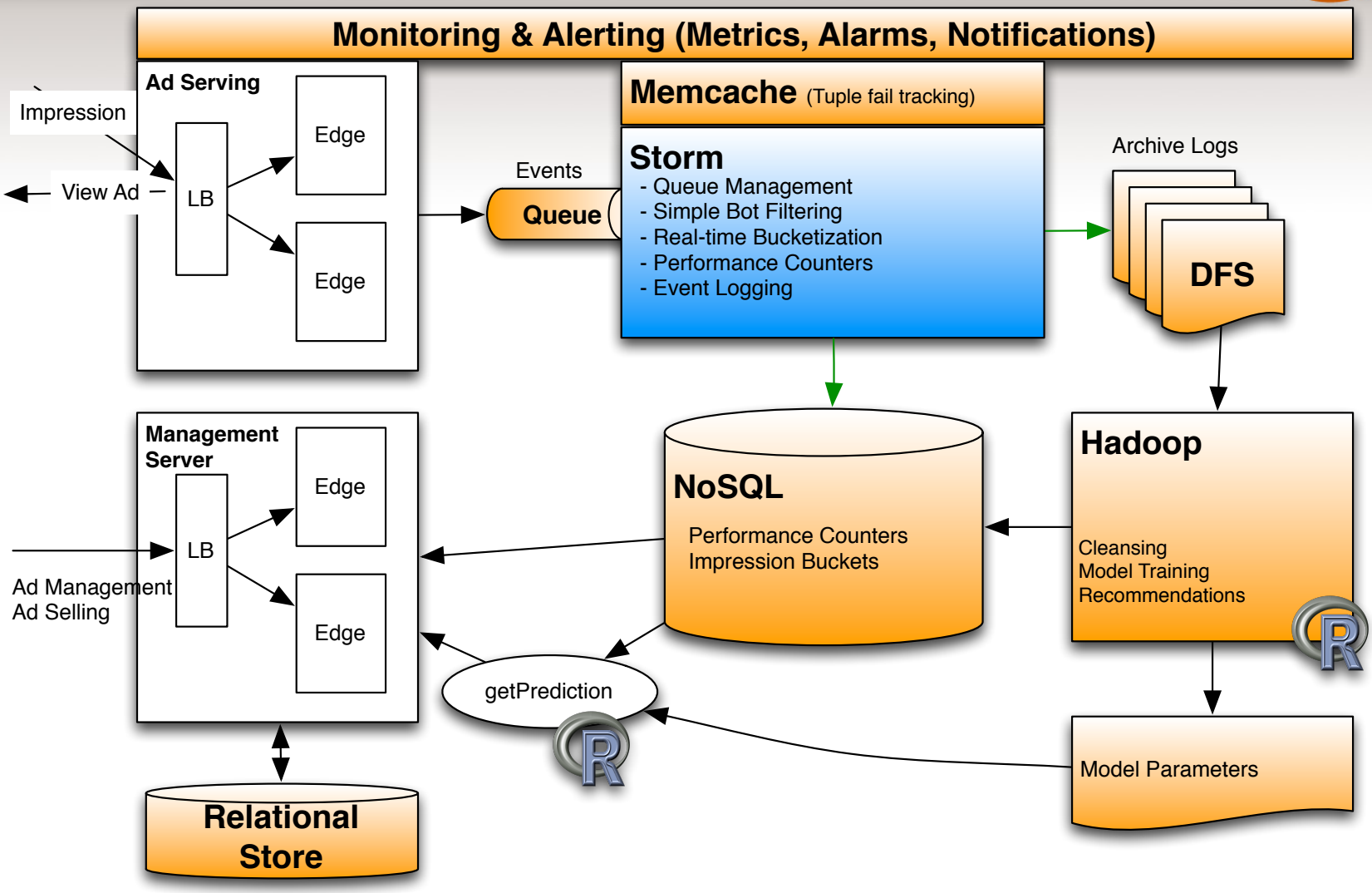
Programming Options

- Raw Java API (tuples and raw computation)
- Trident – Java DSL (SQL-like primitives)
- Python adapter
- Closure DSL
- RedStorm – Ruby
- Wukong – Ruby
- Trident-ML
- Storm-Esper

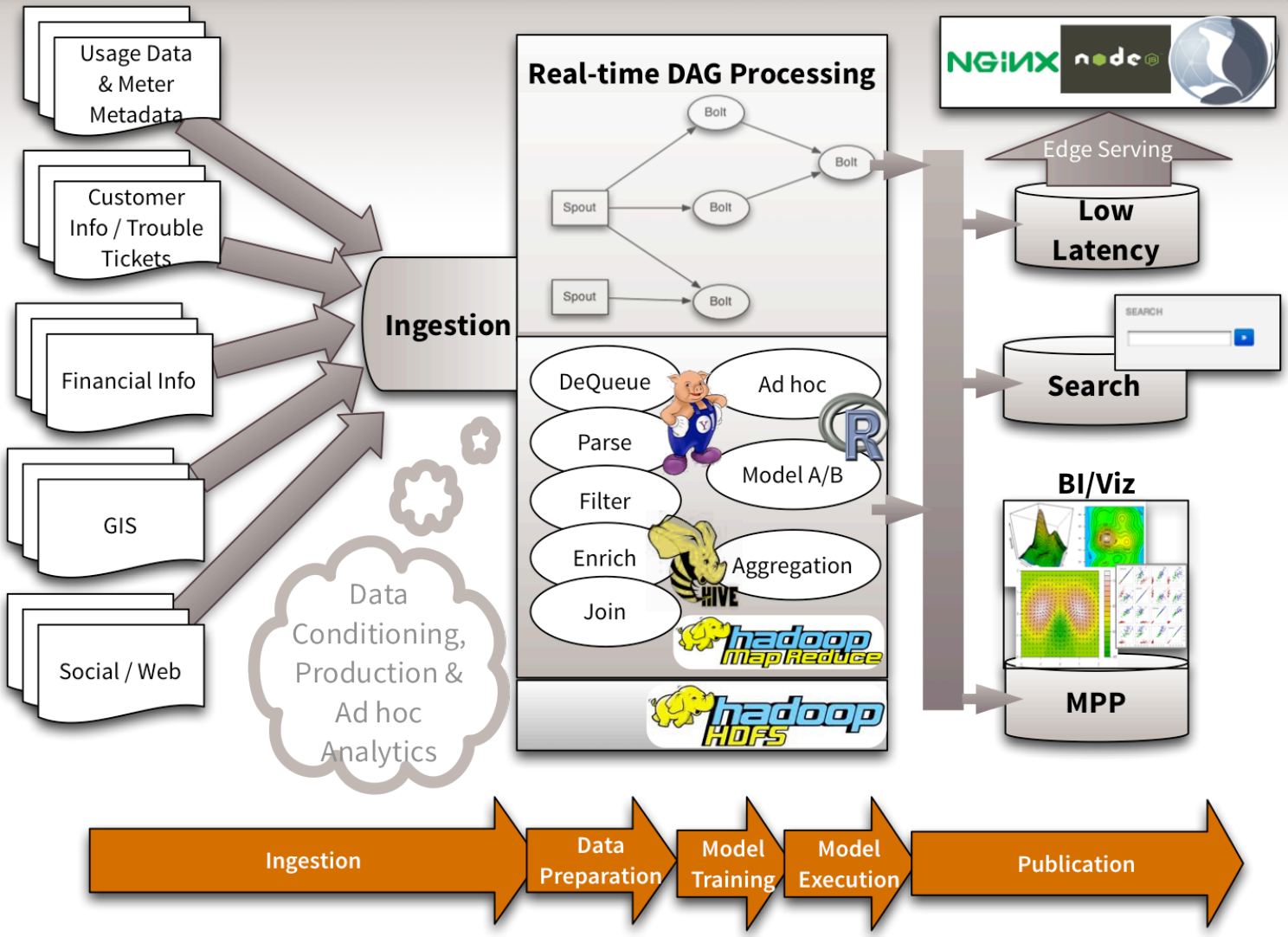
Use Cases

- Scale analysis pipeline
- Lively stats
- Recommendations
- Better predictions
- Realtime analytics
- Online machine learning
- Distributed RPC

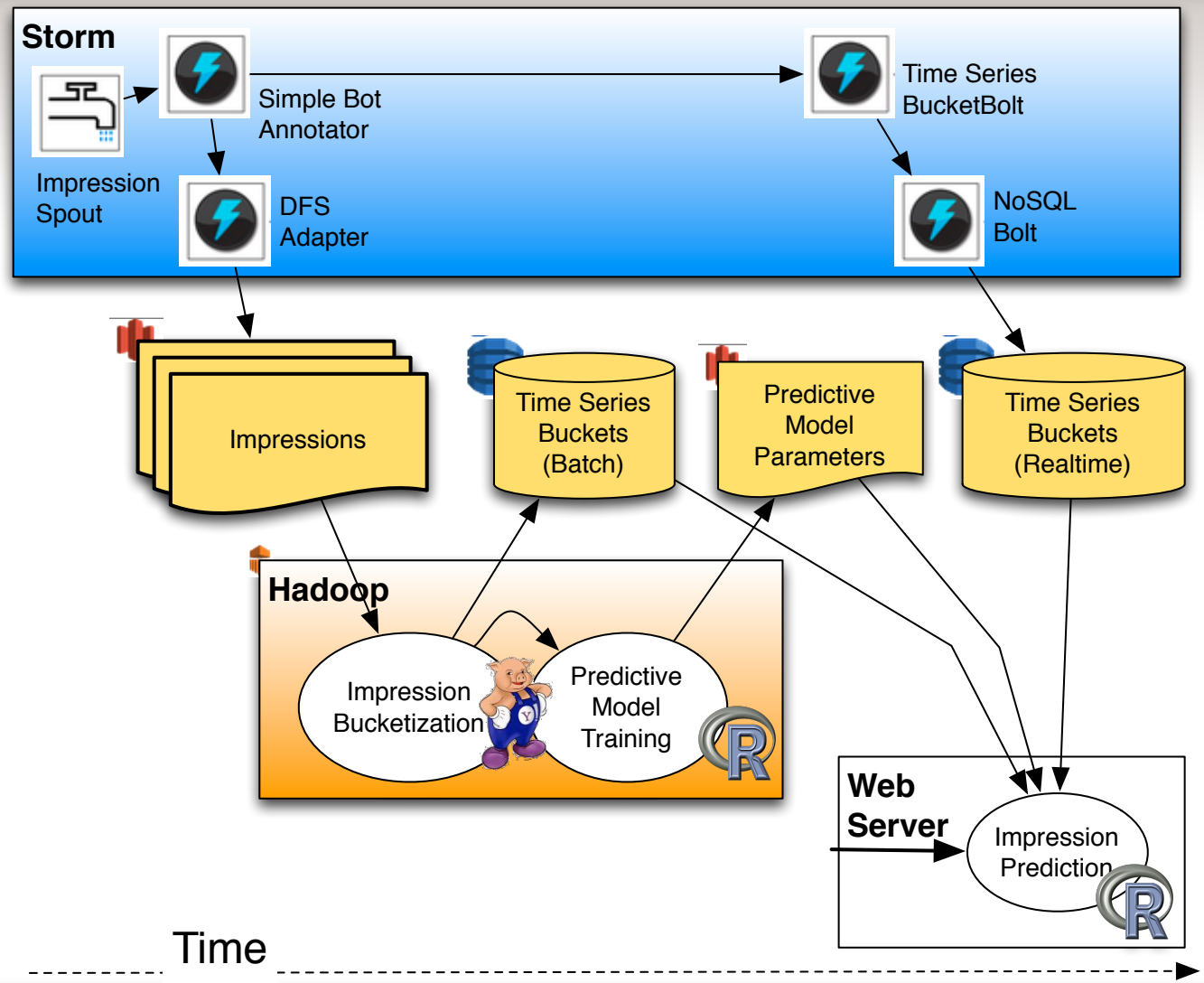
Overall Architecture



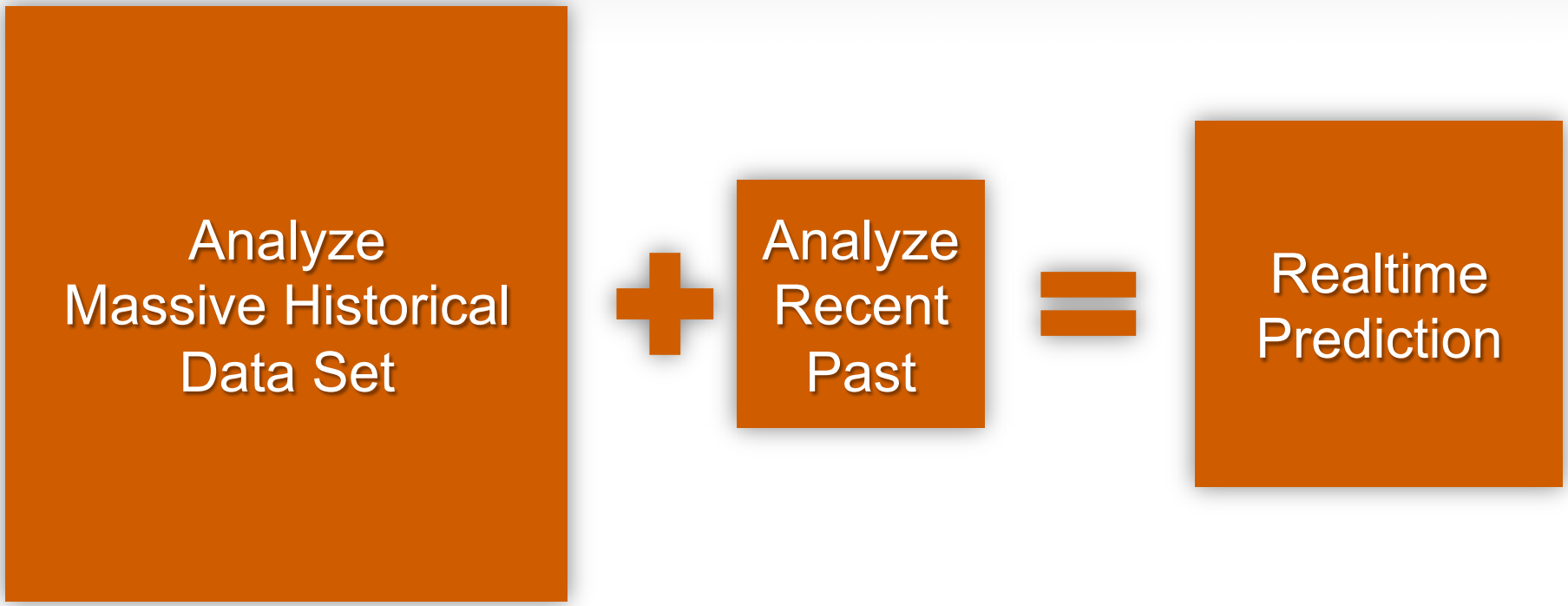
Integration Patterns



Analytics Architecture



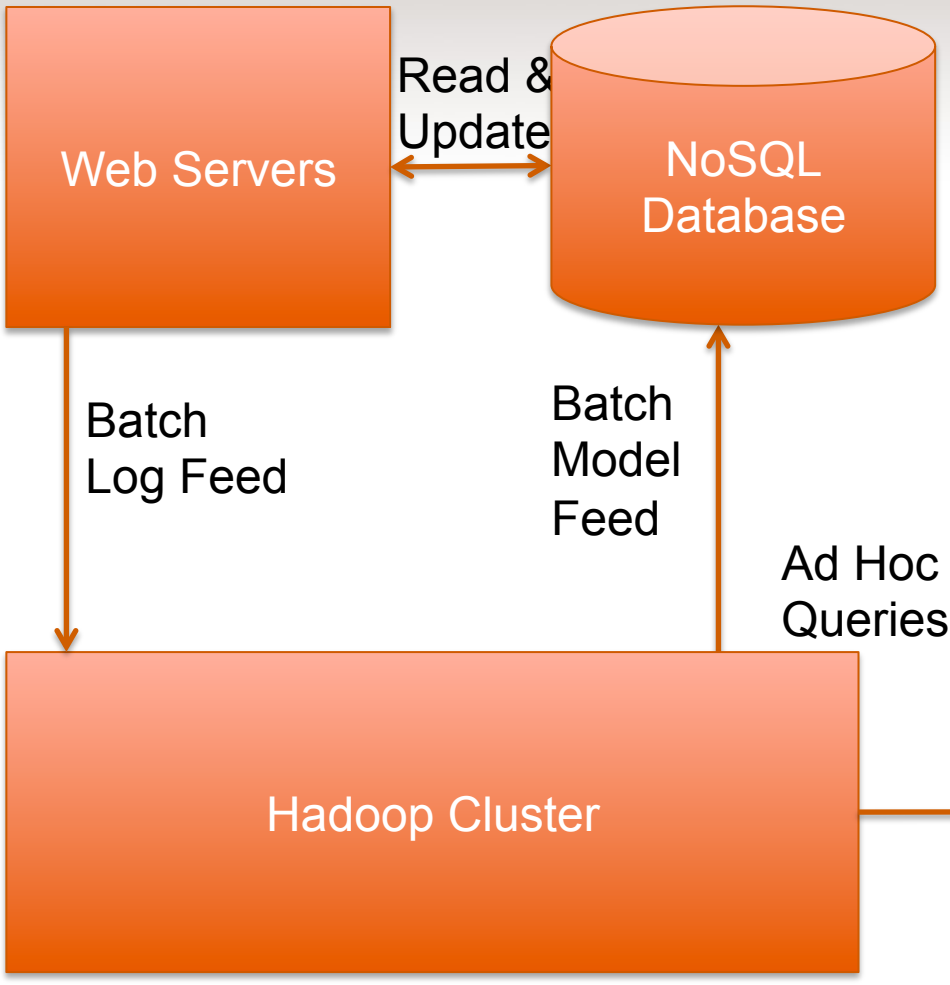
Solution Approach



Historical Data Set = S3
Analyze = Hadoop + Pig + R

Recent Past = Storm + NoSQL
Analyze = R + Web Service

Example: Real-Time Recommendation Updates



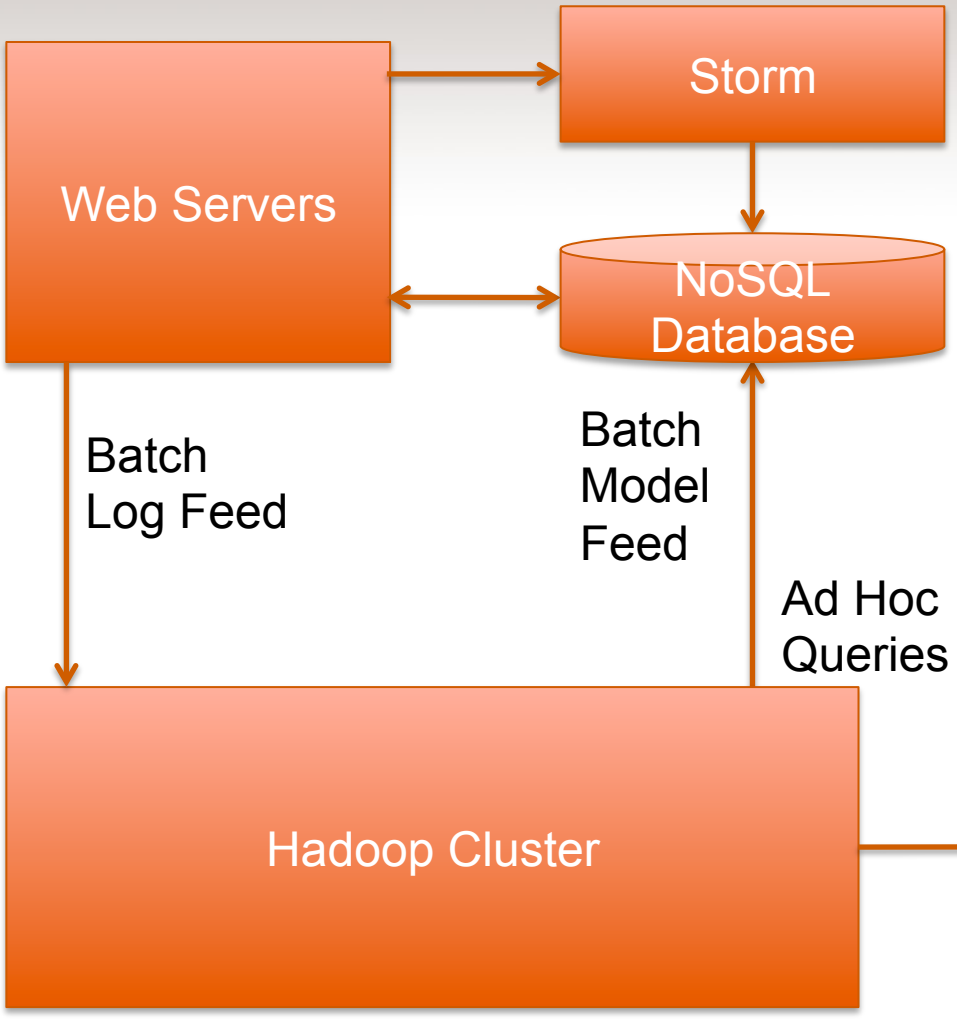
- Requests read single user profile
- May update that profile record
- Classic key/value store problem
- Can split transient state into read/write NoSQL with batch view NoSQL

Classic BI Access

Batch Export

MPP Database

Real-Time Recommendation Updates w/ Storm



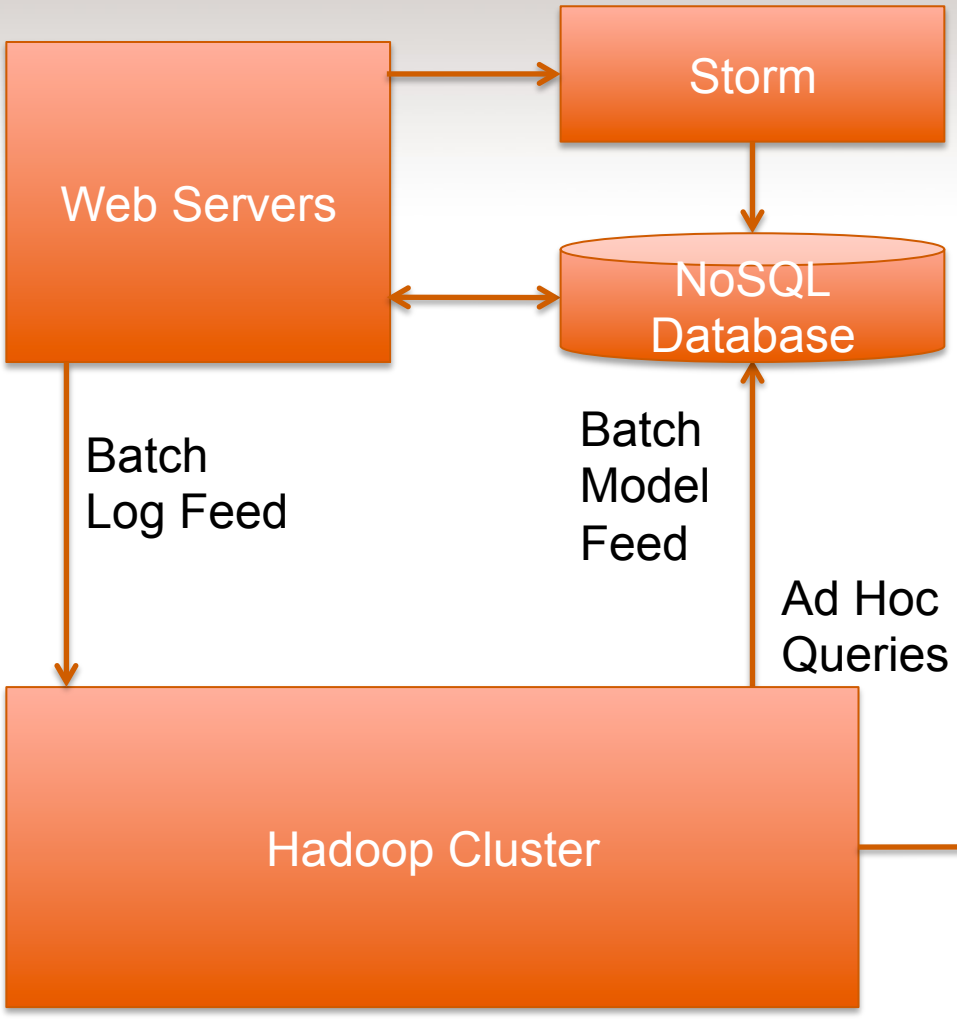
- More complex recommendation logic
- Parallelize analysis across topology
- Processing benefits from distributed logic of streaming

Classic BI Access

Batch Export

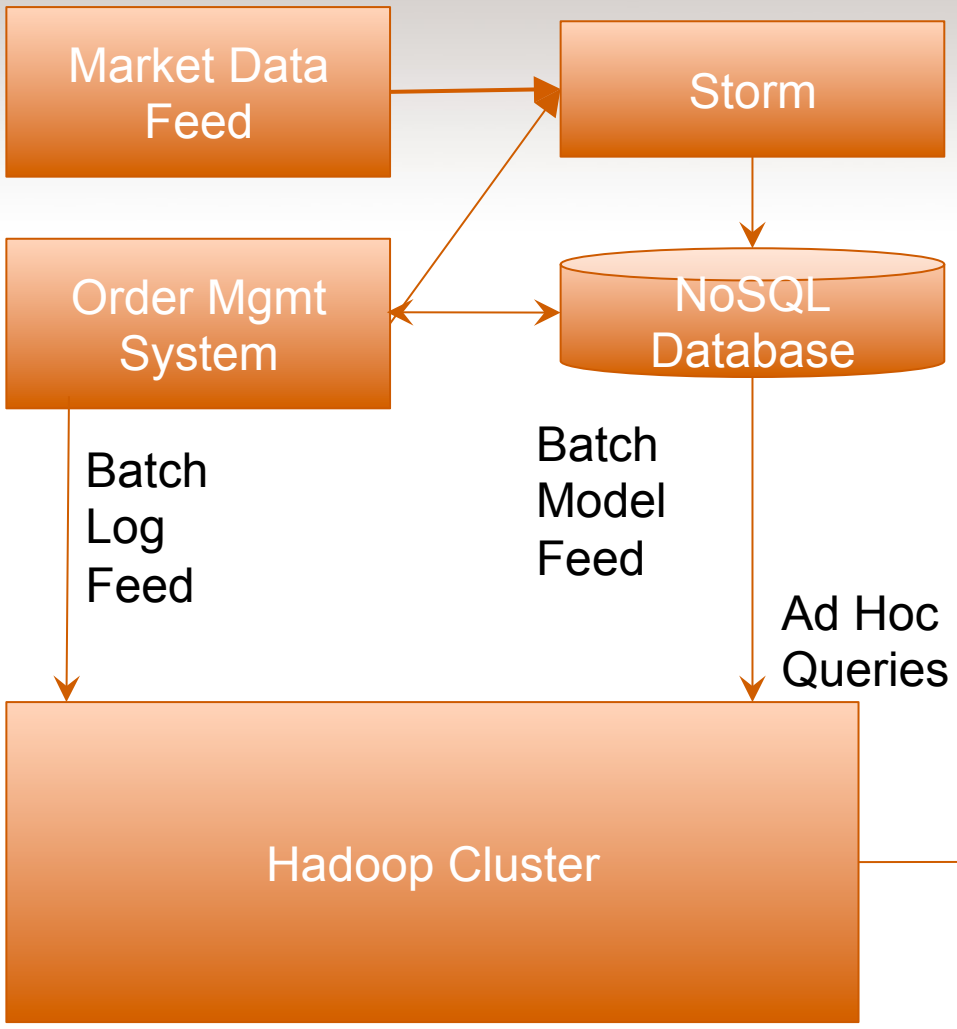
MPP Database

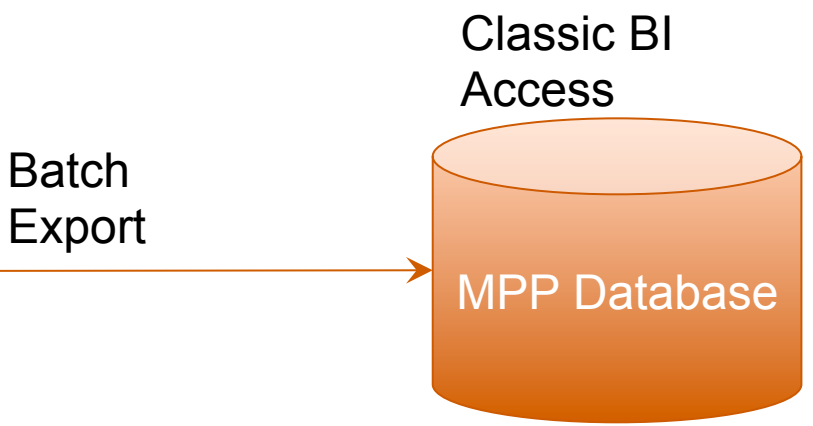
Social Updates (real-time news feed)



- Pages now integrate many profile records
- Fan-out-on-read or Fan-out-on-write
- Processing benefits from distributed logic of streaming

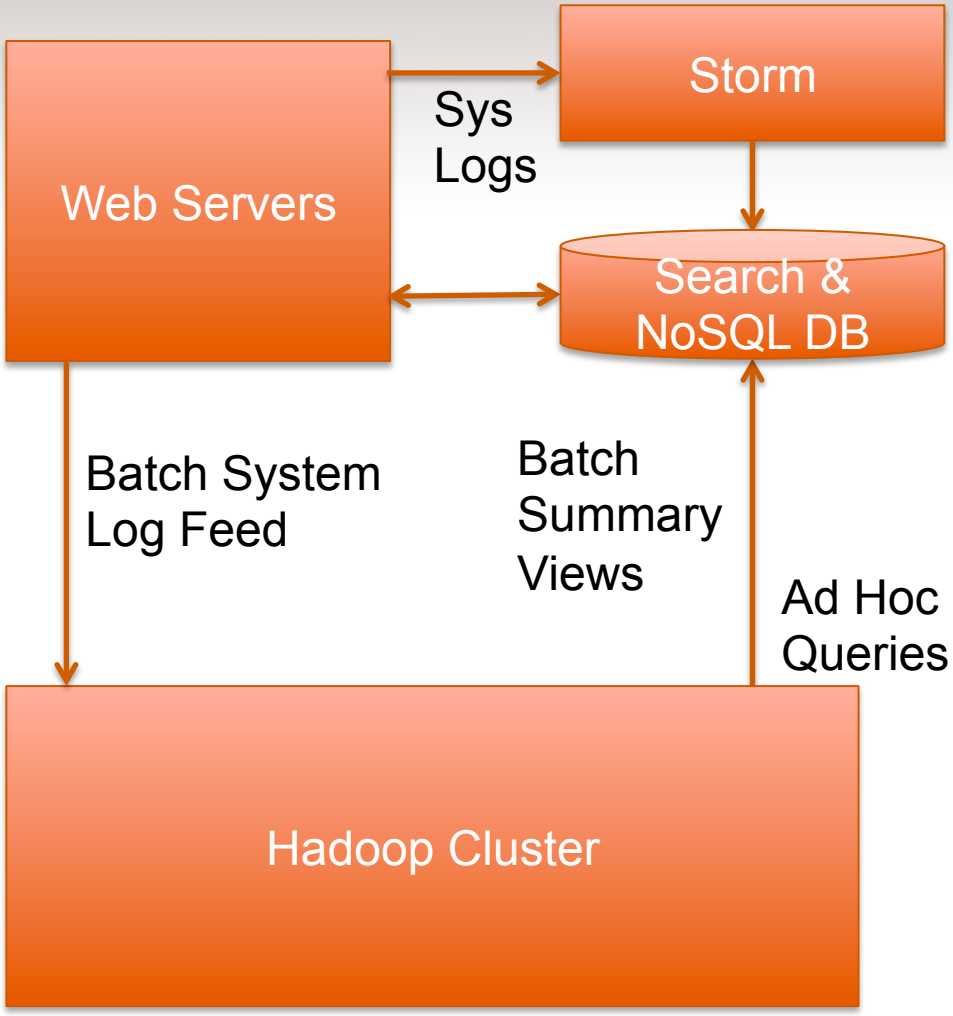
Finance: Trade Compliance Monitoring





- Storm reduces market data to usable subset
- Integrates high, medium and low frequency data
- Real-time Storm filter triggers deeper search via NoSQL

Operational Intelligence: Stats and Search



- Cache and microbatch updates to multidimensional stats
- Support distributed queries
- Cache updates

Lessons

- Easy to develop, hard to debug – start locally
  - Timeouts
- Storm infinite loop of failures
  - Use Memcached to count # tuple failures
- At Least Once processing
  - Hadoop based read-repair job
- Performance Counters not getting flushed
  - Tick Tuples
- Always ACK
- Batching to S3
  - Run a compaction & event de-duplication job in Hadoop

Conclusions

- There's many kinds of real-time problems
- Use of Hadoop and/or NoSQL can solve
  - Low latency queries
  - Event response with localized intelligence
  - Operational intelligence
- Storm is valuable for
  - Ingesting data within seconds
  - Complex real-time distributed logic
  - Operational intelligence

- **We're Hiring!**