Real-time Big Data
Without Streaming

Ron Bodkin
Founder & CEO
Think Big Analytics

purpose built Big Data professional services

@ronbodkin
ron.bodkin@thinkbiganalytics.com
Agenda

• Big Data & Streaming
• Why real-time?
• Evolution of Examples
• Summary
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?

• **Low Latency Analysis**: data exploration, anomaly investigation

• **Event Response**: apply model to adjust behavior – recommendations, content & personalization, alert triage, fraud detection

• **Operational Intelligence**: live dashboards, drill down on dimensions, live reports
Integration Patterns

Source: Think Big Analytics
What is Streaming Big Data?

• A distributed system with
• **Velocity**: Pass data with low latency
• **Volume**: elastic processing and storage
• **Variety**: Flexibility to process diverse data
• **Value?**
Example Big Data Streaming Technologies

- Splunk
- Storm
- S4
- IBM InfoStreams
- SQLStream
- Red Lambda (SIEM)
- Feedzai
- CR-X

- Apache Flume
- Apache Kafka
- Scribe
- Kestrel
- Fuse Fabric
- MapR NFS
- Syslog-ng?
Other Real-Time Big Data Technologies

- Real-Time Query: Big Query, Impala, Platfora, Apache Drill, Hadapt, Splice Machine...
- MPP Databases: Vertica, Greenplum, Netezza, ...
- NoSQL databases: HBase, Cassandra, Druid, MongoDB, ...
- Distributed search: SolrCloud, ElasticSearch, DataStax Enterprise
- Replication & mirroring
- Application Server Clusters
Streaming, but not Big Data

• Traditional message queues
• Single machine Complex Event Processing
• Single machine NFS gateway
Example Scenario: Consumer Website

Web Servers

Batch Log Feed

Hadoop Cluster

Ad Hoc Queries

Batch Export

MPP Database

Classic BI Access
Batch Recommendations

- NoSQL database supports bulk import
- Holds profile data
- Read profile record to render pages

Web Servers ➔ Batch Log Feed ➔ Read ➔ NoSQL Database ➔ Batch Model Feed ➔ Ad Hoc Queries ➔ Batch Export ➔ Hadoop Cluster ➔ MPP Database

Classic BI Access
Faceted Recommendations

- E.g., People You May Know
- Feed model output into distributed search index (e.g., SolrCloud)

Diagram:
- Web Servers
  - Batch Log Feed
- Distributed Search
  - Batch Model Feed
  - Ad Hoc Queries
- Hadoop Cluster
  - Batch Export
- Classic BI Access
- MPP Database
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

Web Servers

NoSQL Database(s?)

Batch Log Feed

Batch Model Feed

Hadoop Cluster

Batch Export

Classic BI Access

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

Diagram:

- Web Servers
- Streaming Servers
- NoSQL Database
- Hadoop Cluster
- Batch Log Feed
- Batch Model Feed
- Ad Hoc Queries
- Classic BI Access
- MPP Database
- Batch Export
Operational Intelligence: Metrics Dash

- Each server updates recent statistics
- Blend batch views and transient updates (or just keep transient)
Operational Intelligence: Search

- Maintain search indexes
- Fan-out logic to collect search results
- Data platforms provide more or less out of the box
- Processing benefits from distributed logic of streaming
Operational Intelligence Package

- Products like Splunk
- Dashboards
- Search
- Alerts

Web Servers → Indexing & Aggregation Layer

Batch System Log Feed → Batch Summary Views → Ad Hoc Queries

Sys Logs

Hadoop Cluster
Low Latency Analytics

Web Servers

Streaming Ingestion

Fast Query Engine

Hadoop Cluster
Low Latency Analytics

• Fast emerging space for real-time query
• Fast *response* to queries doesn’t need streaming
• But streaming is relevant depending on data *recency* in query
• Micro-batch data ingestion: 5+ minute latency
  – Sqoop, Chukwa, Informatica, Pentaho, Talend, Attunity, custom scripts
• Queries against very recent data fed by
  – *streamed* data ingestion into Hadoop: seconds+
  – *replication*: seconds+
  – reading the *source system*, e.g., Cassandra, HBase in serving cluster: milliseconds+
  – *streamed* queries to distributed storage merged with batch data: milliseconds+
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
• Streaming is valuable for
  – Ingesting data within seconds
  – Complex real-time distributed logic
  – Operational intelligence