PRESENTED BY RED HAT

LEARN. NETWORK.
EXPERIENCE OPEN SOURCE.

Large Scale/Big Data Federation & Virtualization: A Case Study

Vamsi Chemitiganti, Chief Solution Architect

Derrick Kittler, Senior Solution Architect

Bill Kemp, Senior Solution Architect

Red Hat

06.29.12


Orioles
FILA

Red Hat Big Data – Week In Retrospective

- Big Data, Volume, Speed & Benefits with Red Hat JBoss Data Grid
- Red Hat's Big Data Strategy Overview & Optimizing Apache Hadoop with Red Hat Enterprise MRG Grid
- JBoss Enterprise Middleware & Big Data
- NoSQL & Big Data at Red Hat
- Large Scale / Big Data Federation & Virtualization: A Case Study

Goals

- “Big” Data by Example
- Significant Learning/s
- Solution Architectures

Background – Big Data

- Data growing at 40% per year (McKinsey Global Institute) – a lot of analyst reports exist...
- Big Data
  - Data size and performance requirements become significant design and decision factors for implementing a data management and analysis system.
  - In this definition, there's not an absolute size milestone between "data" and "big data."
- 4V's
  - How much? formats? speed? change?

Background – Where is it coming from ?

- Volume - “old” and “new” types of data
  - transaction volumes and other traditional data types
- Variety – more types of information to analyze
  - social media, mobile (context-aware)
    - tabular data (databases), hierarchical data, documents, e-mail, metering data, video, still images, audio, stock ticker data, financial transactions, etc...
- Velocity – how fast is data produced?
  - how fast must the data be processed?
- Variability - data unpredictability, new forms, risk !
  - UPC barcodes, RFID scanners, sensors (HVAC), etc...

Background – Big Data Tech Landscape

- NoSQL (Not only SQL)
- Brewers CAP Theorem
  - Cassandra, MongoDB, Neo4J, Hive, Pig, JDG, etc...
- Data Federation and Virtualization
  - JBoss Enterprise Data Services Platform
- MapReduce and batch processing
- And the list goes on ...

Infinispan

mongoDB

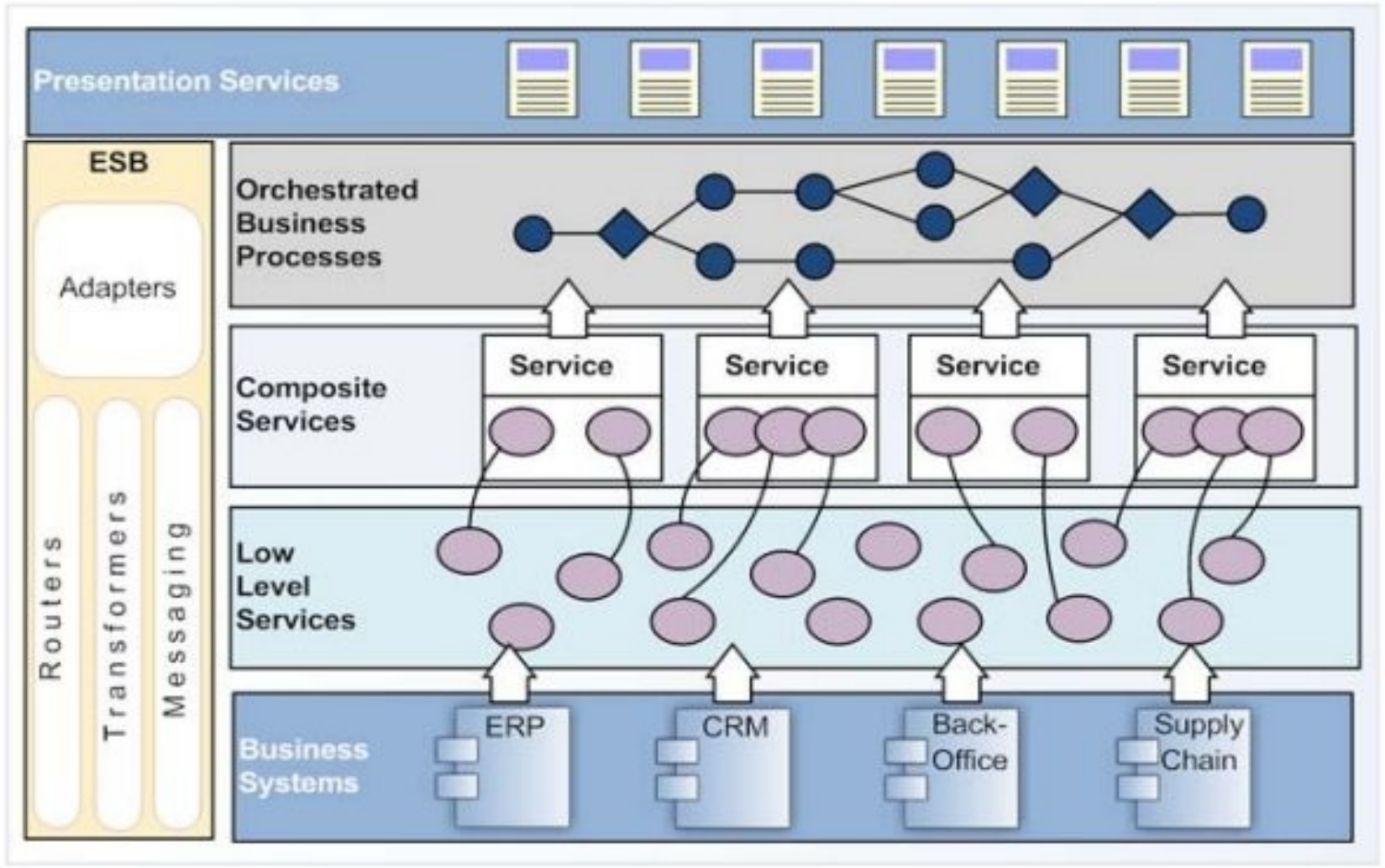
Background – Financial Services Industry

- Large commercial bank
  - History of large acquisitions
    - Different sources of data that capture only parts of their overall data lifecycle
- Fast moving business and compliance environment
  - Dodd-Frank and Basel 2 regulations
- Need agility on top of all their data challenges

The Problem – Case Study Problem Domain

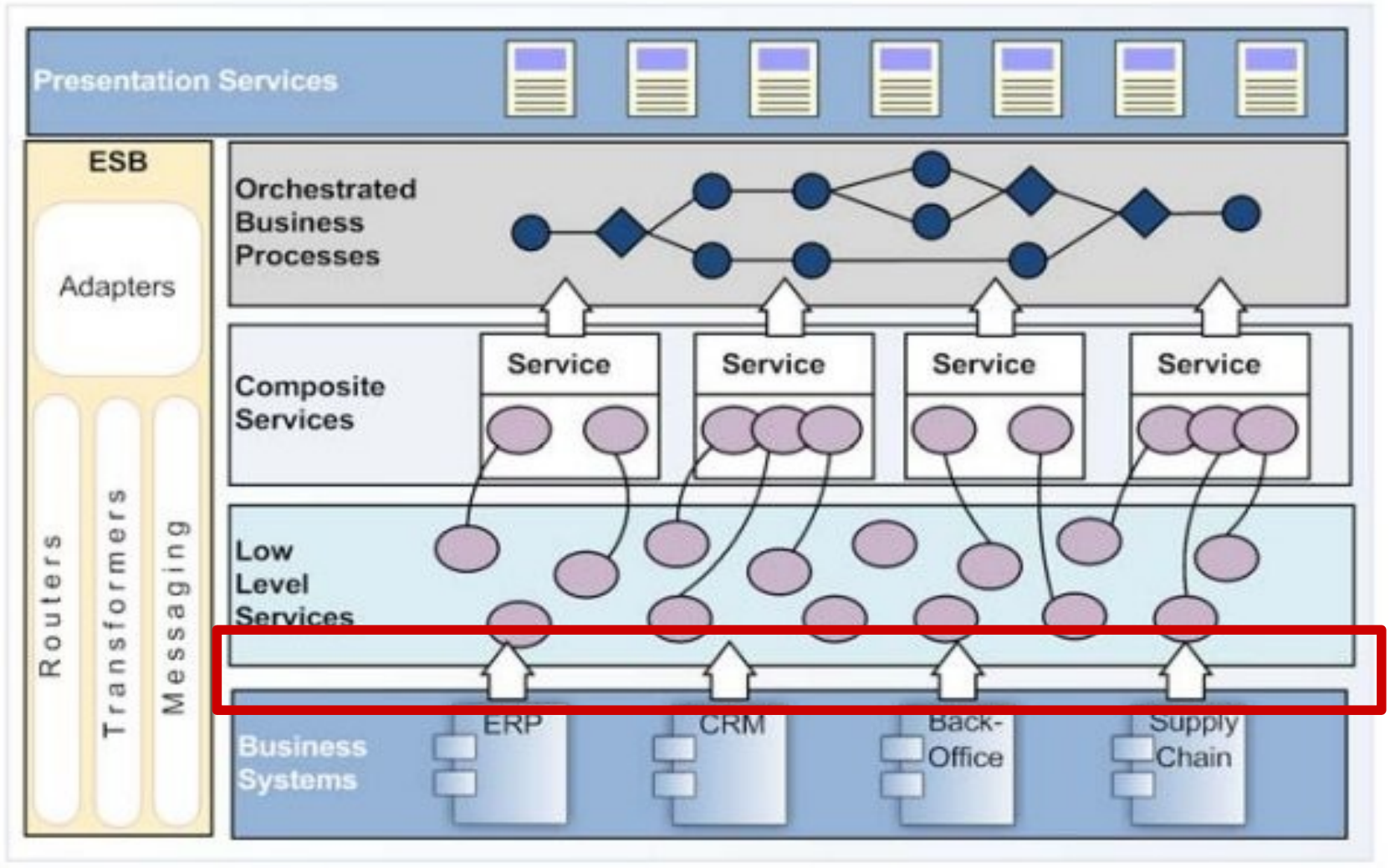
- Acquisitions and partner integration
  - Many sources of financial data with different origins and formats.
- Primary Business Drivers
  - Business Intelligence (predictive analytics and forecasting)
  - How to harness the volumes of Data; 'Big' Data
  - Compliance and Risk Management
  - Provide exposure to clients via Multiple Channels
  - Large pains around ETL

Reference Architecture



Open Source SOA, Jeff Davis, Manning Press

Reference Architecture

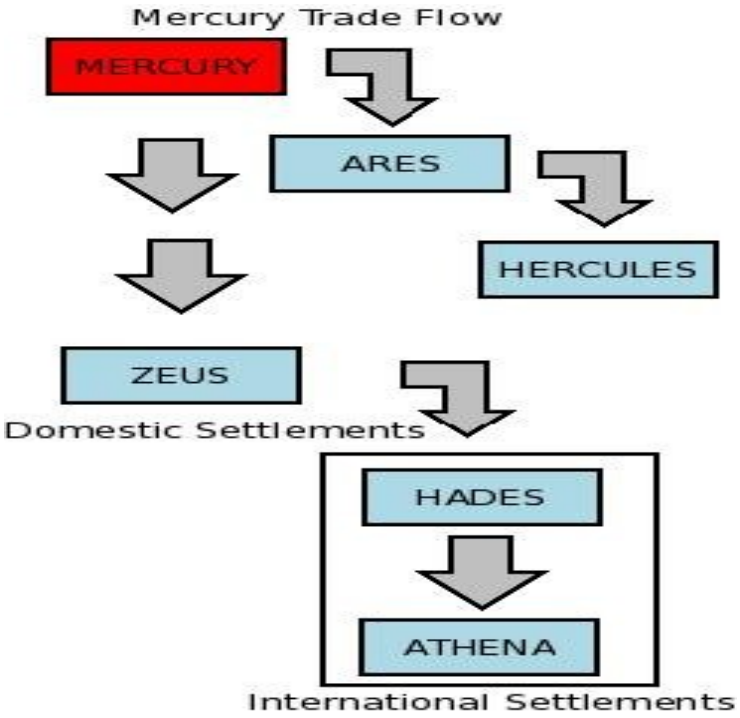


Open Source SOA, Jeff Davis, Manning Press

The Problem – Data sources

App Name	Format	Size	# Table
MERCURY	MF – ASCII Extract	~ 450 GB	1 tbl/day
ATLAS	M/F ASCII Extract	~ 54 GB	1 tbl/day
ARES	Oracle DB	~ 350 GB	8-10 tbls
HERCULES	Oracle DB	~ 200 GB	11-12 tbls
APOLLO	ASCII File	~ 10 GB	1 tbl
ZEUS	ASCII File	~ 10 GB	1 tbl
ATHENA	Oracle DB	~ 20 GB	8-10 tbls
HADES	Sybase Table	~ 50 GB	8 tbls
HERA	XML Dump	~ 10GB	9 tbls
DEMETER	Oracle DB	~ 10 GB	10-12 tbls

The Problem – Overall Data Flow

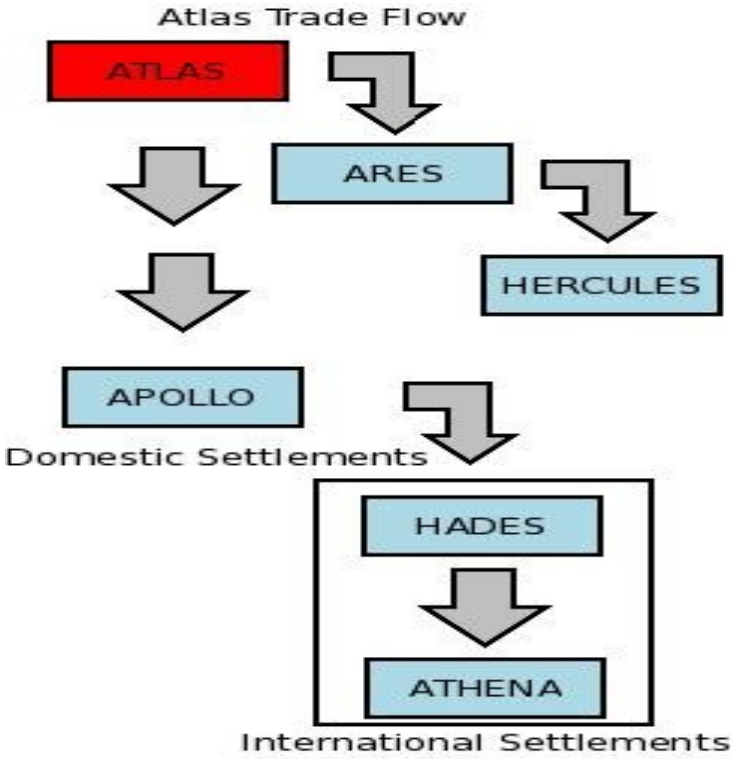


Mercury Trades

- Data *from Mercury flows through to ZEUS*
- *If international data – ZEUS pass to International Settlement systems*
- *Data from MERCURY is also logged to Trademart (ARES) system*
- *Position information are captured in the GPDW (HERCULES) application*

ATLAS Data

- *Trades from ATLAS flows through to APOLLO (domestic trade)*
- *If international data – ATLAS passes to ARES and then to HADES/ATHENA*
- *Position information are captured in the GPDW (HERCULES) application*



The Problem – views of data...

- Trade View
  - Rationalization, data agility, data source flexibility
  - Easily and rapidly augment models with new sources and/or data attributes
- Account View
  - Federate across multiple account sources and create a virtualized canonical mode
  - Data augmentation, federation and data source flexibility
    - Service real time data integration challenges
- Instrument View
  - Rationalization, virtualization, data source flexibility

The Problem – data hierarchies

Trade Source Data systems

Priority	Source
1	ARES – Trade DM
2	HERCULES
3	APOLLO
4	ATHENA
5	HADES

Account Source Data systems

Priority	Source
1	AMC Master
2	ARES – Trade DM
3	HERCULES
4	APOLLO
5	ATHENA
6	HADES

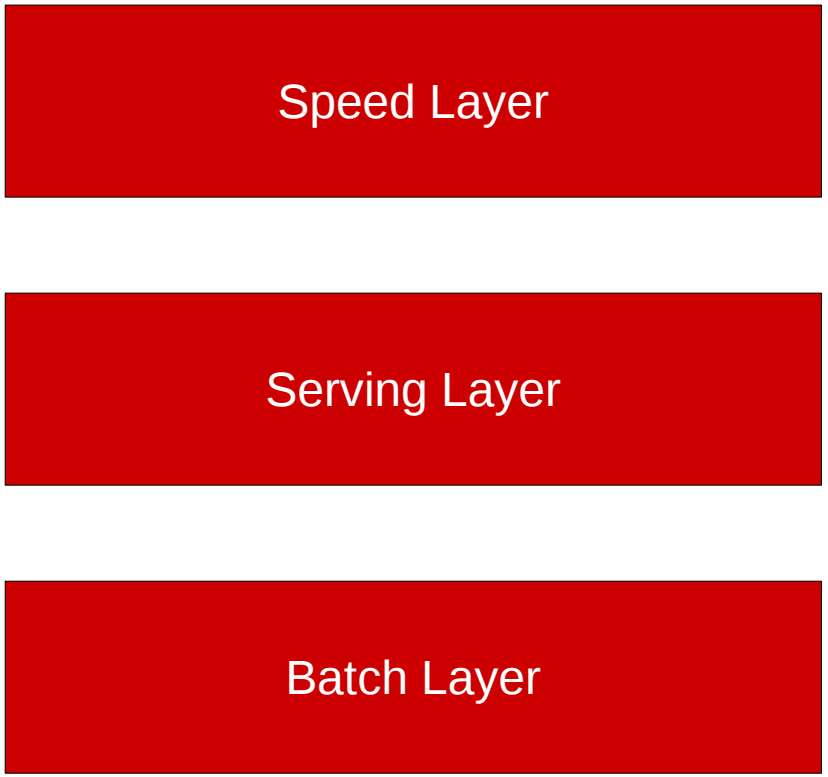
Instrument Source Data systems

Priority	Source
1	SMC Master
2	ARES – Trade DM
3	HERCULES
4	APOLLO
5	ATHENA
6	HADES

Significant Learnings – not as easy as you think

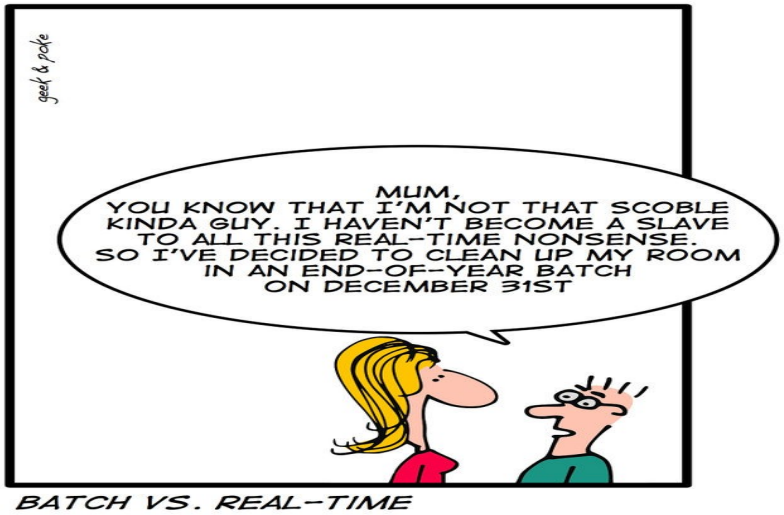
- Source database performance matters
  - Database tuning is important for push-down model
- Materialization is a read-only solution
  - Write back is disabled!
- Unstructured data is not query-ready
  - Pre-processing, indexing and optimization is required
- Data virtualization provides the ability to discover nuances in the source data and relationships
  - Learn more about your data

Solution



The Solution – Batch Layer

- High Latency
  - Lots of data
    - Continual compute
- Pre-compute Views of Data
  - Master data set / all data
  - Constantly growing !
- MapReduce is a Canonical Example
  - Red Hat Storage, Hadoop, Etc...




```
private static void runBatchLayer() {
        while( true ){
                recomputeBatchViews();
        }
}
```

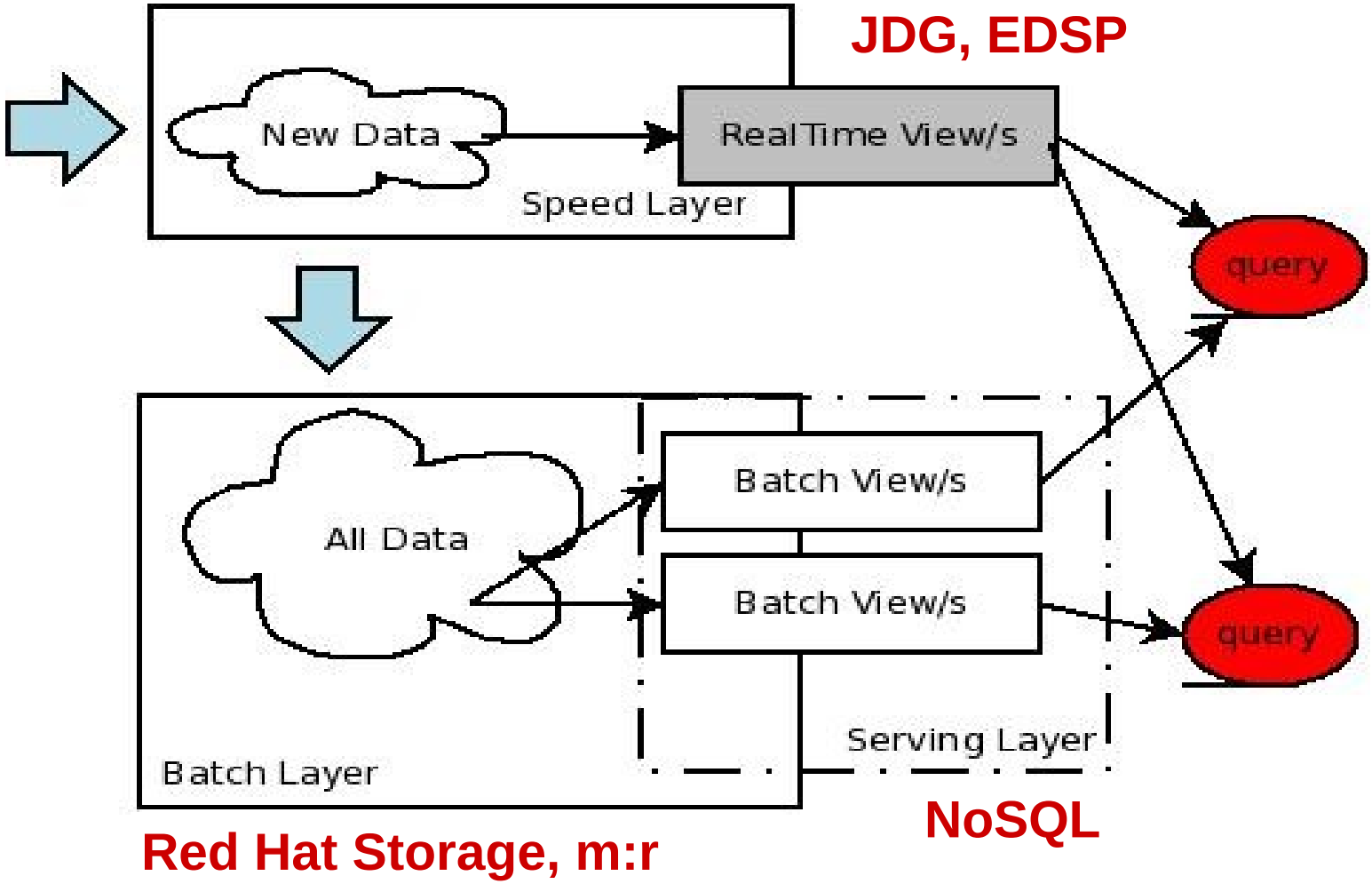
The Solution – Serving Layer

- Loads the batch views
  - Indexes for efficient querying
    - Continual compute
- Pre-compute Views of Data
  - Master data set / all data
  - Constantly growing !
- NoSQL is a Canonical Example
  - MongoDB, Cassandra, Neo4J, Etc...
    - Key-Value, Graph, Document, Column??

The Solution – Speed Layer

- Similar to Service Layer but Faster !
  - Doesn't look at all new data at once
  - Updates real-time view as it receives new data
- Incremental Updates vs. Re-computation Updates
  - Produces views only on **RECENT** data vs. entire dataset
  - Random reads/writes
  - Way more complex than batch and serving layer
- Data Federation/Virtualization, Caching, etc..
  - Enterprise Data Services Platform, JBoss Data Grid

Solution Architecture

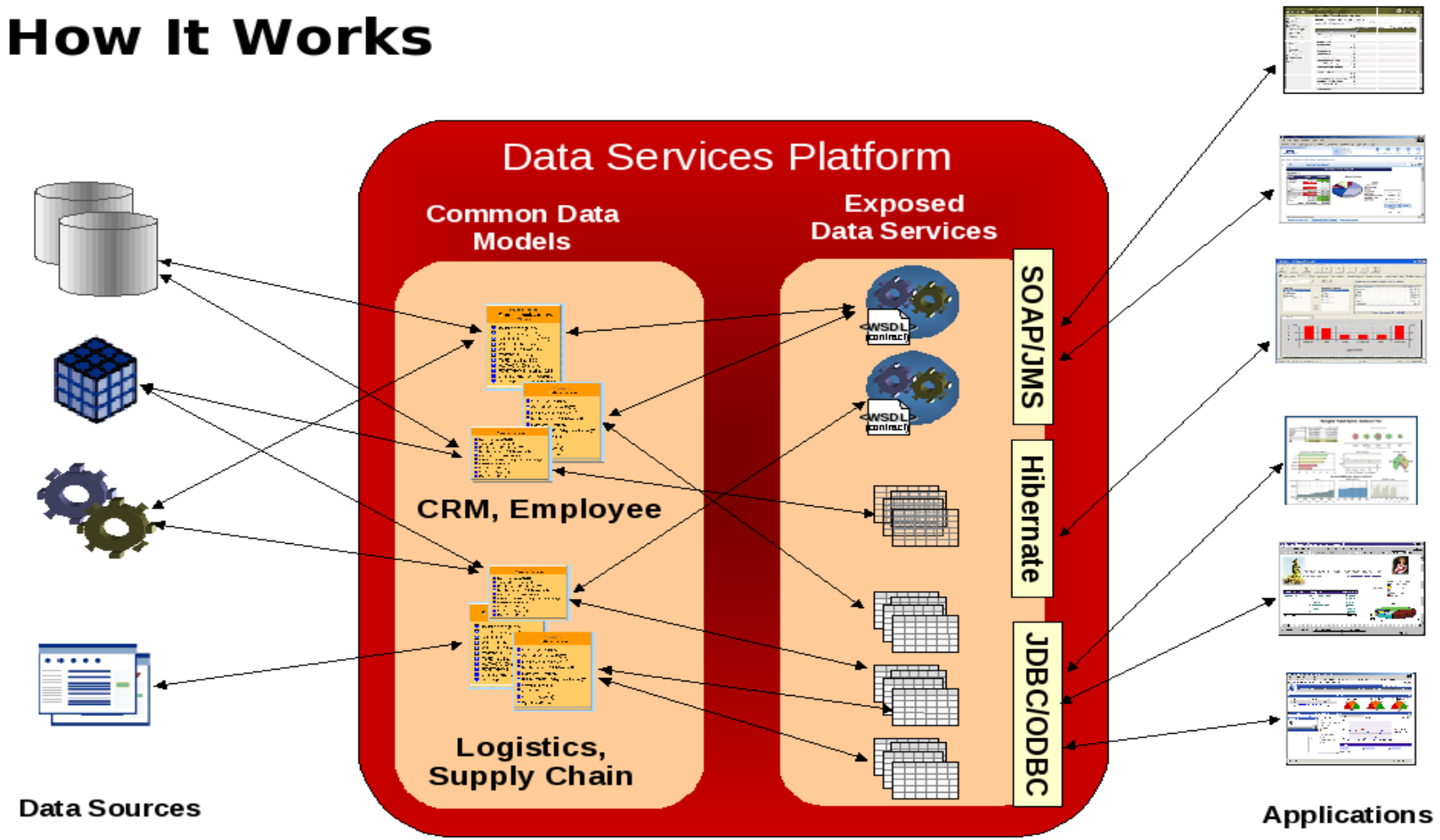


Big Data, Nathan Martz, Manning Press

Solution Architecture

- Enterprise Data Services Platform
  - Speed layer; real time data and batch data access
- JBoss Data Grid
  - Speed layer and Service layer; in-memory data
- Cassandra
  - Built for analytics; fast writes, highly consistent
    - Maintains indexes for Speed layer
- MapReduce/Red Hat Storage
  - Continual processing of master data
    - Many jobs / continual processing

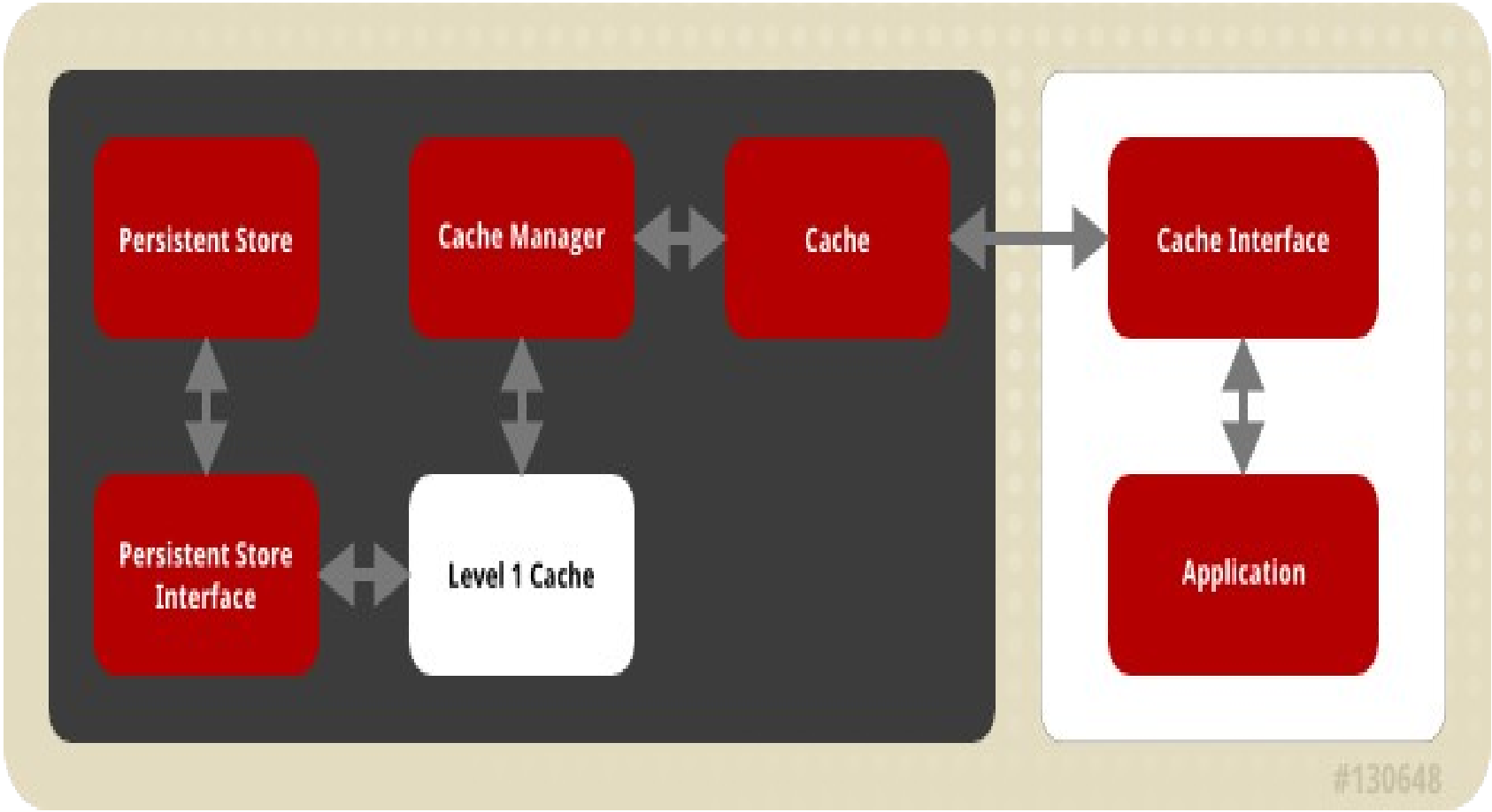
Enterprise Data Services Platform



Red Hat Storage – applicability

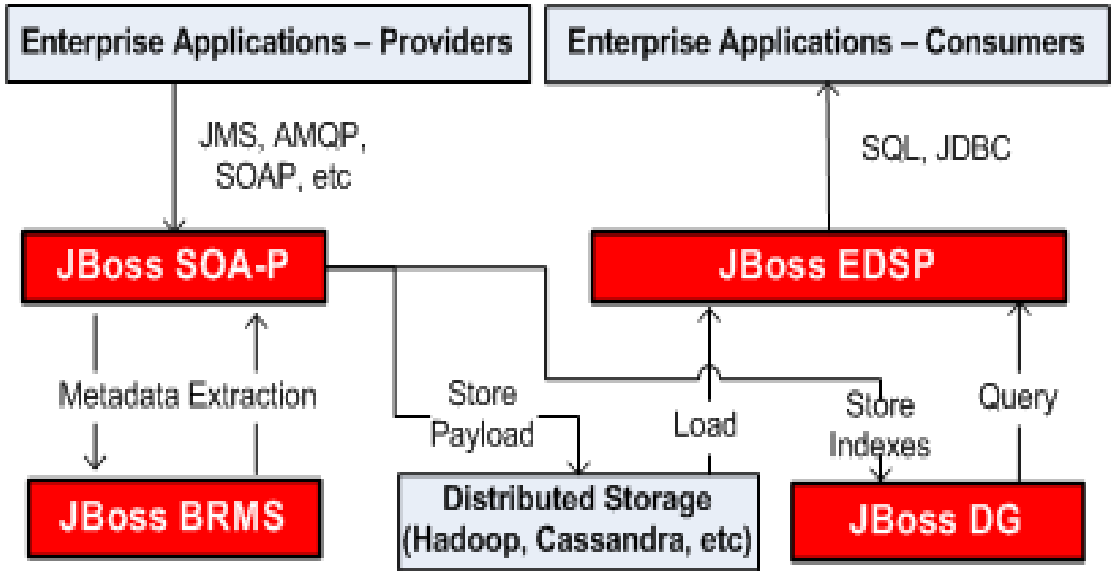
- Out-of-box compatible with MapReduce apps
- Superior Storage Economics
- NAS, NFS, CIFS, HTTP Access to data
- Unify Data Storage
- Eliminate need for NameNode

JBoss Data Grid – Core Architecture



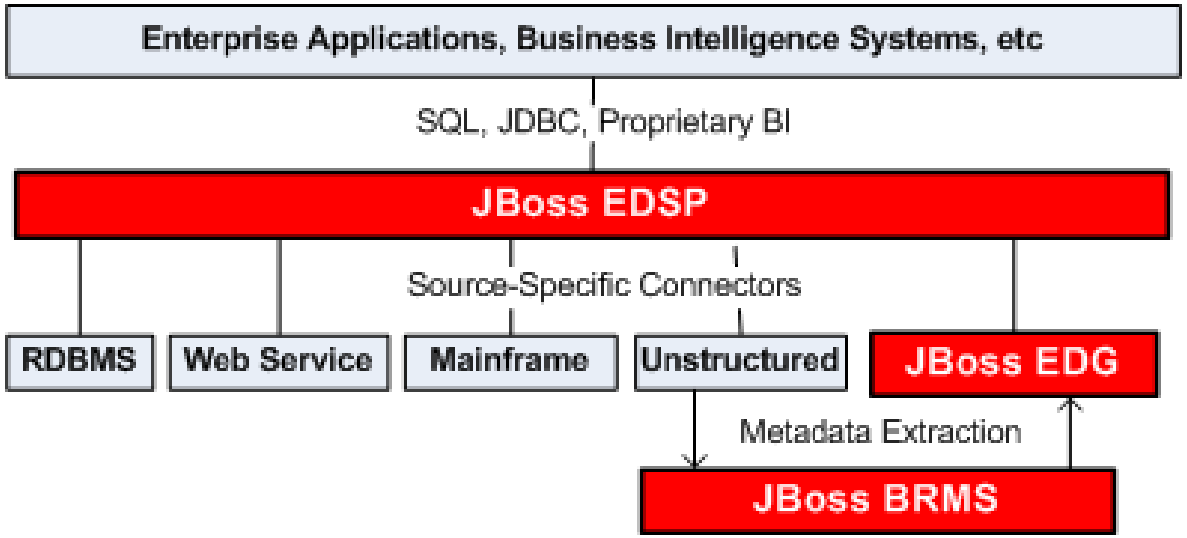
Improving Integration of Big Data into Enterprise Application Architectures

- Red Hat's Solution
  - Existing data producers , standards-based interfaces into the ETL pipeline.



Achieving Greater Transparency into Enterprise Data and Big Data Assets

- Red Hat's Solution
  - A virtualized view of enterprise data, regardless of the source or type. It copes with large amounts of unstructured data via an ETL intake pipeline that extracts and store metadata in a fully customizable manner, increasing overall data visibility.



QUESTIONS?

LIKE US ON FACEBOOK

www.facebook.com/redhatinc

FOLLOW US ON TWITTER

www.twitter.com/redhatsummit

TWEET ABOUT IT

#redhat

READ THE BLOG

summitblog.redhat.com

GIVE US FEEDBACK

www.redhat.com/summit/survey