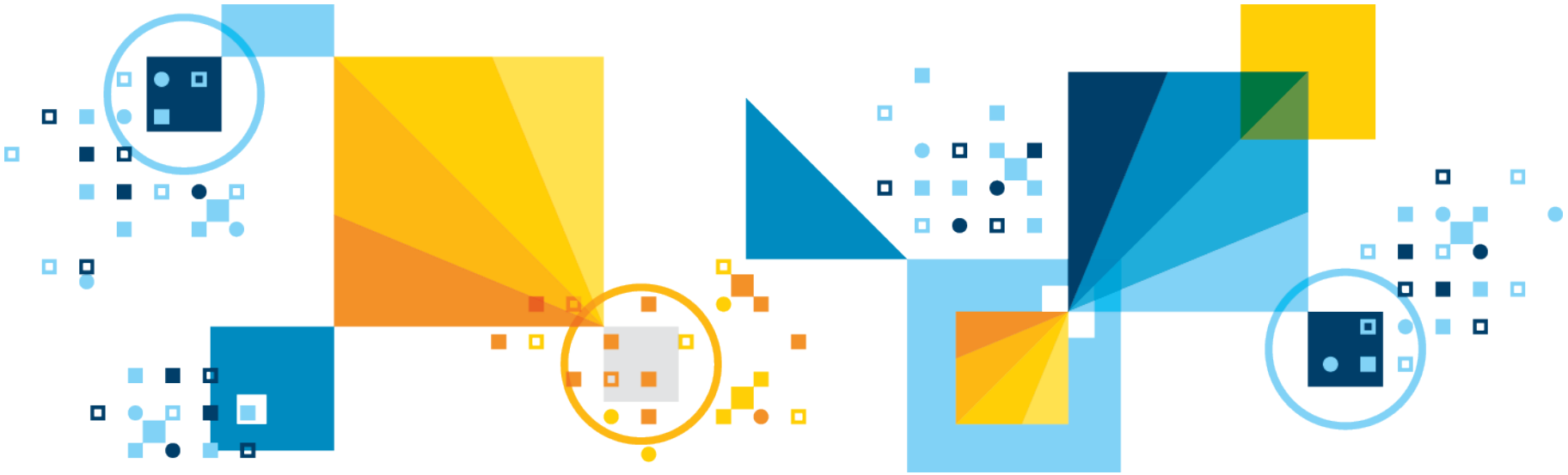
Analyzing and Searching Streams of Social Media

Using Spark, Kafka, and Elasticsearch

Markus Lorch, mlorch@de.ibm.com, @MarkusLorch



Outline

- **Introduction and Scope**
  - IBM and Twitter Partnership
  - IBM Insights for Twitter on IBM Bluemix

- Technology and Experiences
  - Apache Spark in Streaming Mode as the Processing Engine
  - Apache Kafka as a distributed Messaging Queue
  - Elasticsearch as an “Index-based Repository”
  - Hardware Hosted on IBM Softlayer

Introduction and Scope

Related: TECH, DEAL

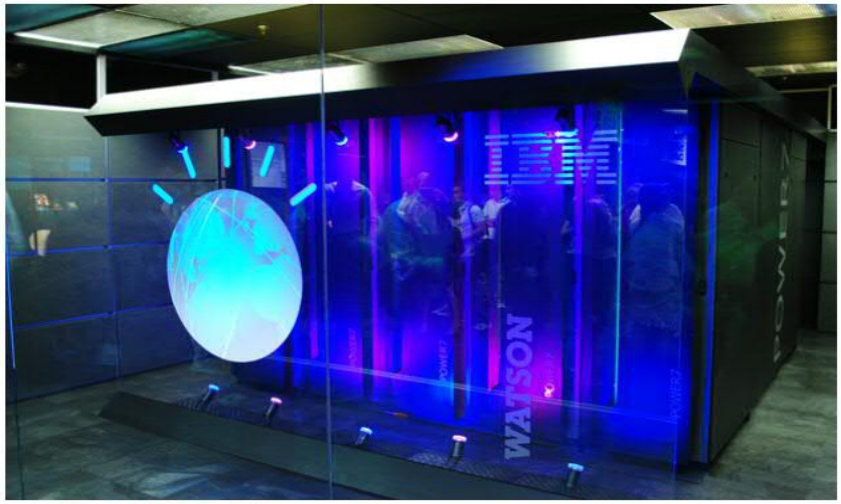
Technology | Wed Oct 29, 2014 4:27pm EDT

IBM, Twitter to partner on business data analytics

WASHINGTON | BY MARINA LOPES

BIG DATA

IBM Introduces Twitter-Fueled Data Services for Business

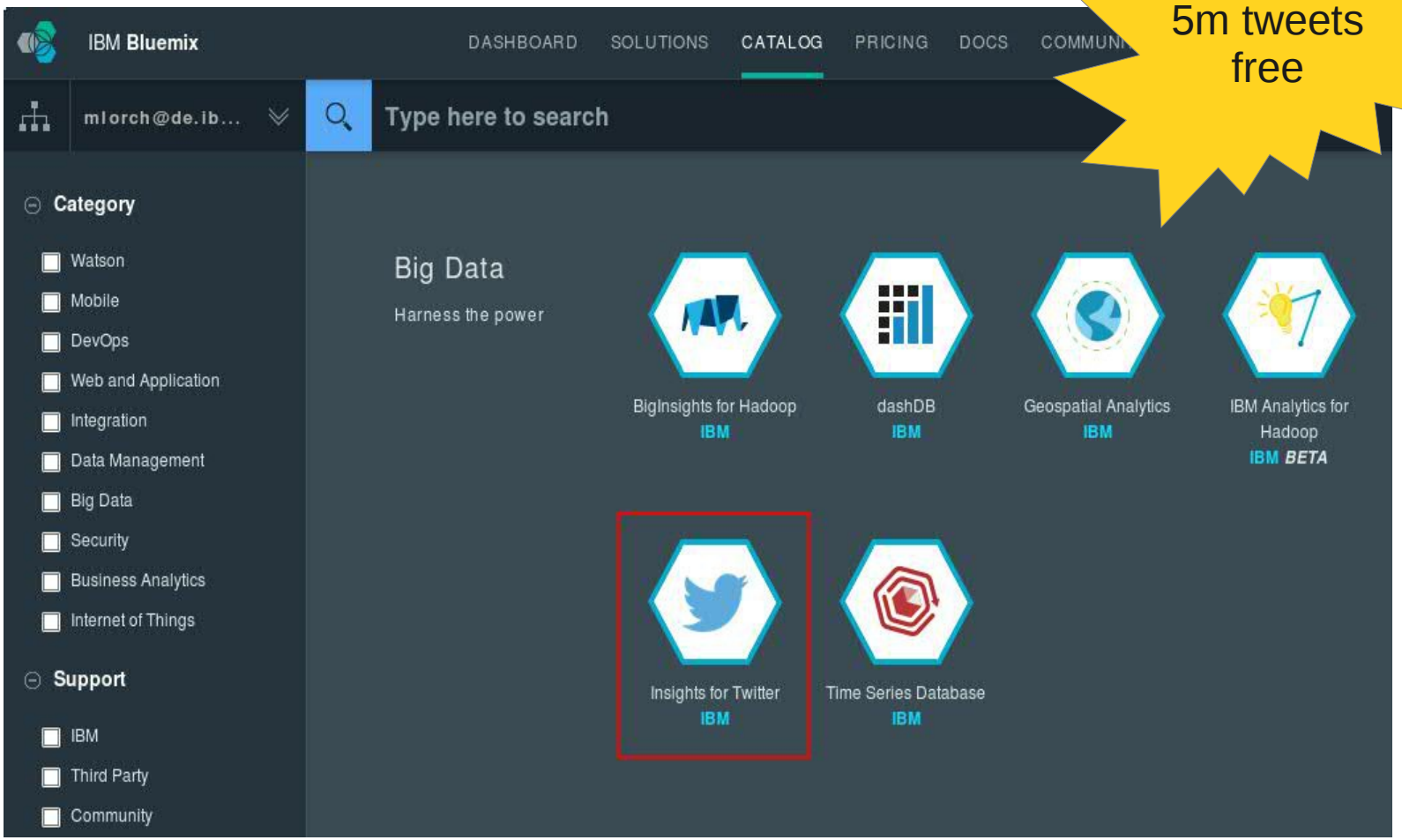


Credit: Wikimedia/Clockready

IBM's new Insights service harvests data from millions of tweets and uses Watson to analyze them for sentiment and behavior

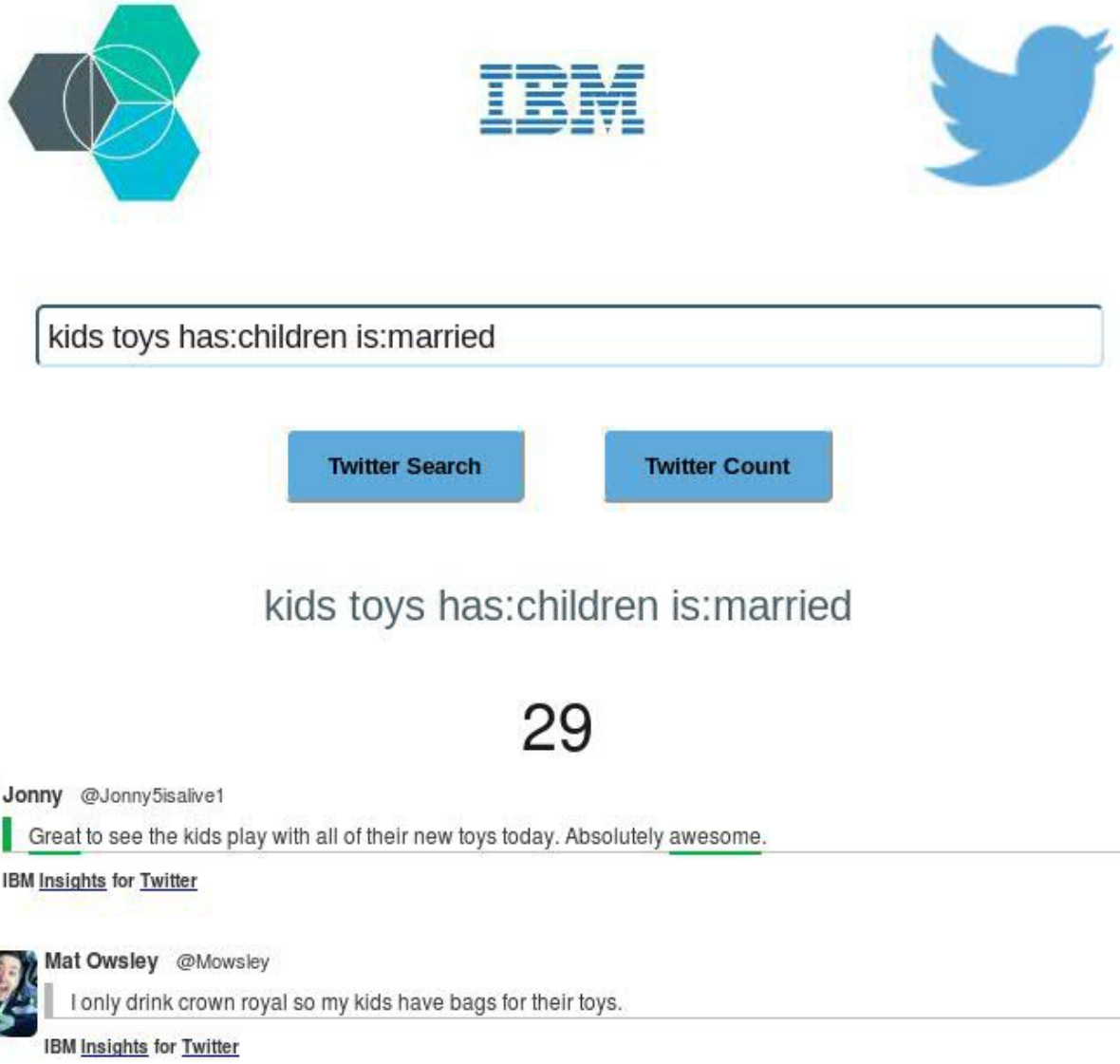
IBM Watson mines Twitter for sentiments

Insights for Twitter Service on IBM Bluemix

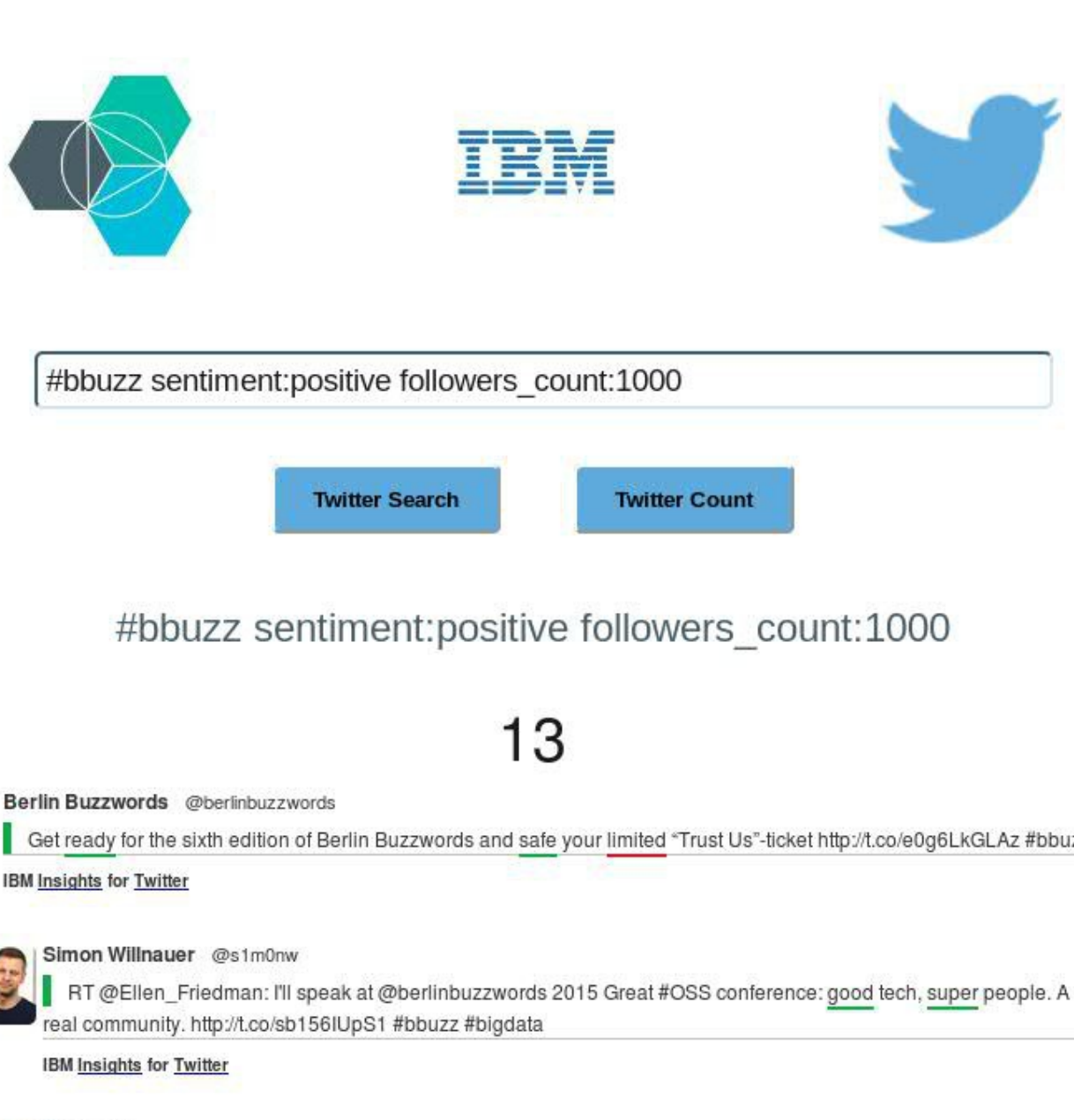


Use it to build your own service leveraging Twitter Tweets and IBM Analytics

Sample Application



#BBUZZ EXAMPLE

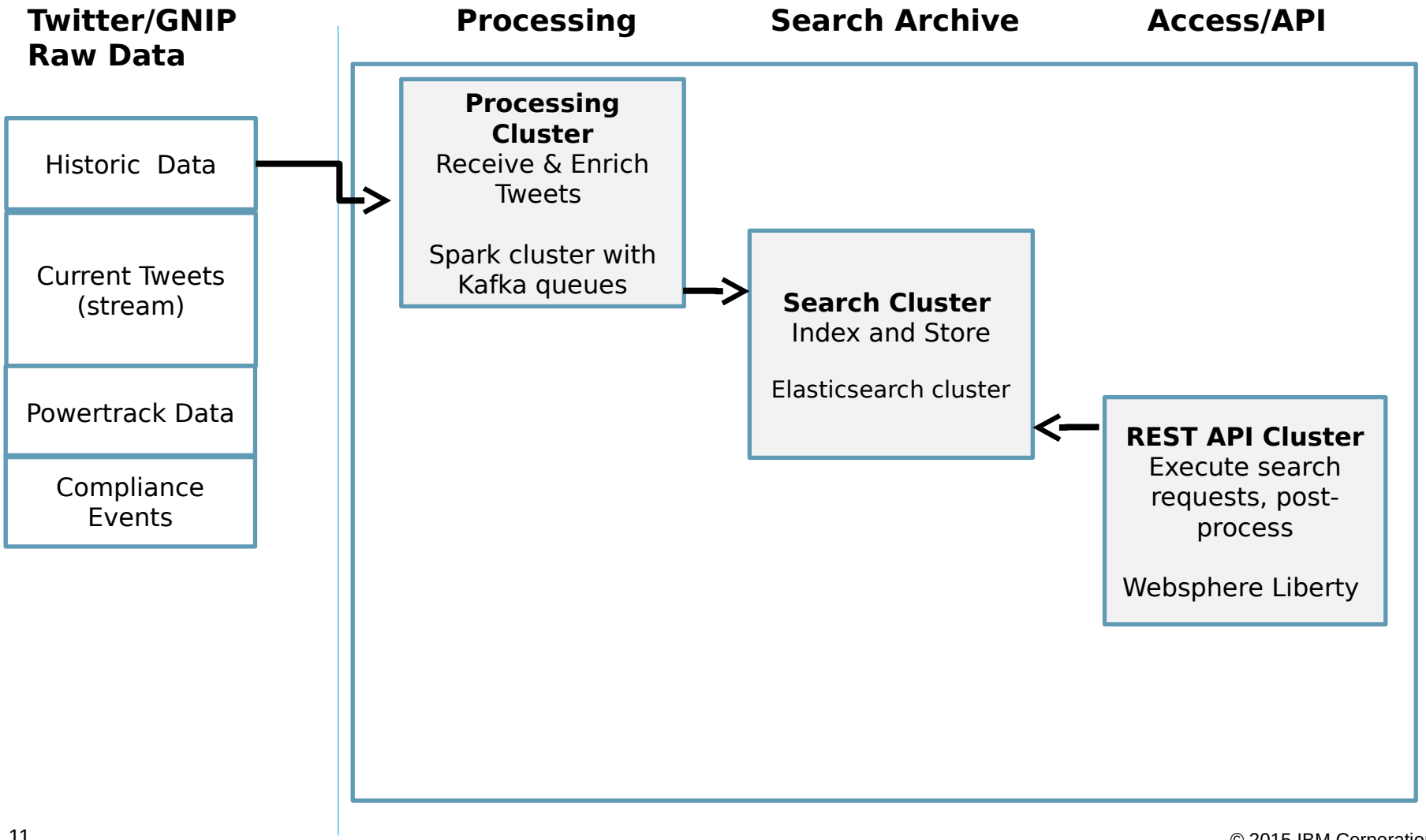


Outline

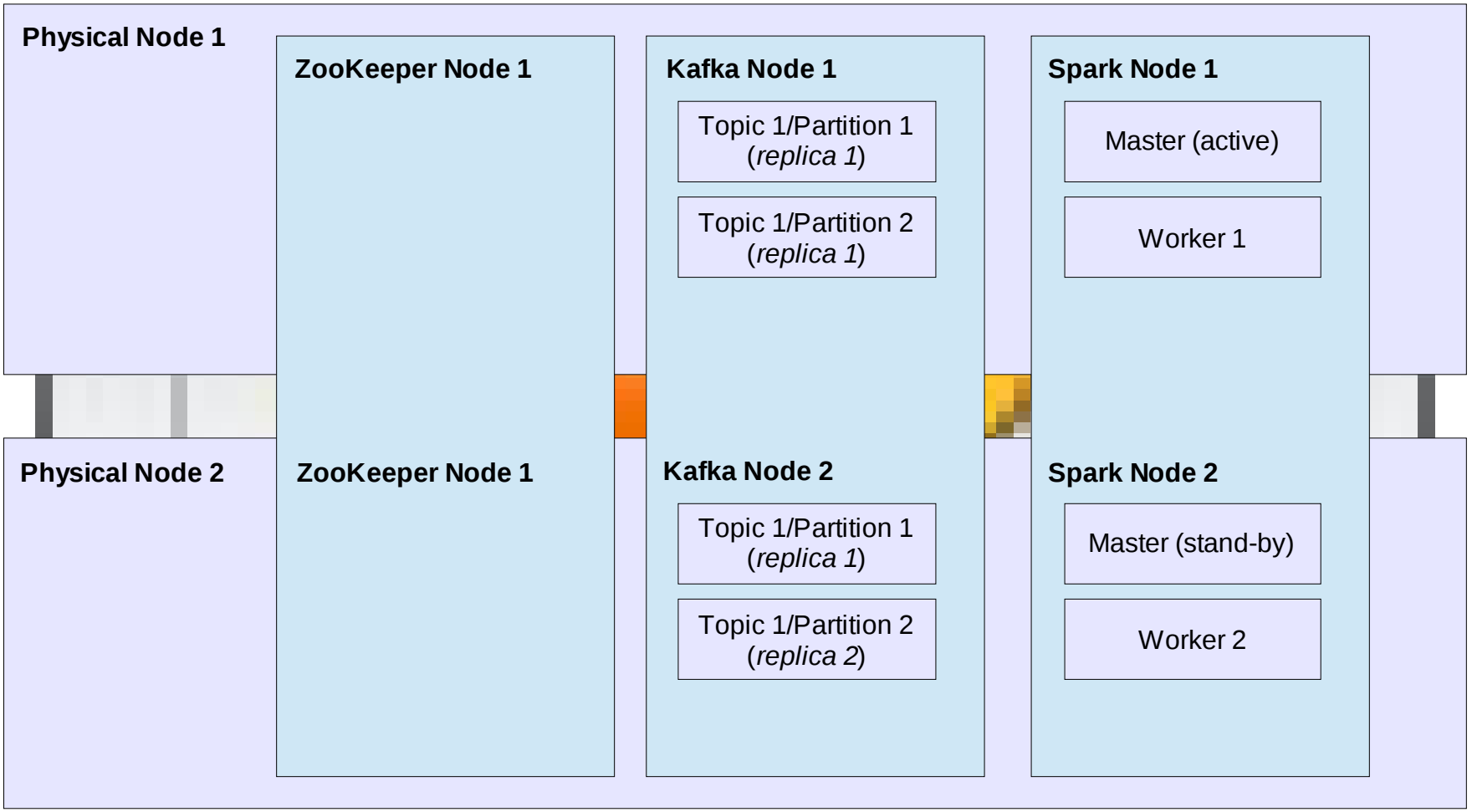
- Introduction and Scope
  - IBM and Twitter Partnership
  - IBM Insights for Twitter on IBM Bluemix

- **Technology and Experiences**
  - Apache Spark in Streaming Mode as the Processing Engine
  - Apache Kafka as a distributed Messaging Queue
  - Elasticsearch as an “Index-based Repository”
  - Hardware Hosted on IBM Softlayer

High-level Architecture



Use of Kafka and Spark



System Design

Hardware

- System running on IBM Softlayer bare-metal servers, use many (relatively) small servers which no hardware redundancy.
- Smaller servers → faster recovery from failure and higher redundancy
- each function has a minimum of 3 servers to ensure HA even in the case of maintenance
- Continuous availability (rolling restarts)

Software

- All redundancy provided by software stack
- Use Spark as processing engine
- leverage Spark streaming with micro-batching
  - future: direct streams with better Kafka integration
- Use Kafka as distributed messaging / queueing system with message persistence
- Leveraging a large Elasticsearch cluster as an index-based repository optimized for low query-time
- Leveraging IBM Websphere Liberty for REST API implementation

Experiences

- Kafka helps to decouple processing and queue messages
  –> ability to delay incoming processing
- Kafka also allows us to read raw-data as well as analyzed data with multiple consumers (e.g. index but also write to files)
- Spark streaming with micro-batching adds about 1 sec delay, creates very small RDDs
- Spark streaming causes inefficient copying of data from Kafka, and issues with locally stored RDDs (spark scratch)
- Existing analytics code (java) was easy to get to run in Spark, much new analytics code is being written for Spark
- Elasticsearch provided a solid and scalable search engine, but with larger cluster size maintenance is not effortless
- Storing the Tweets only in the index avoids joins with DB storage