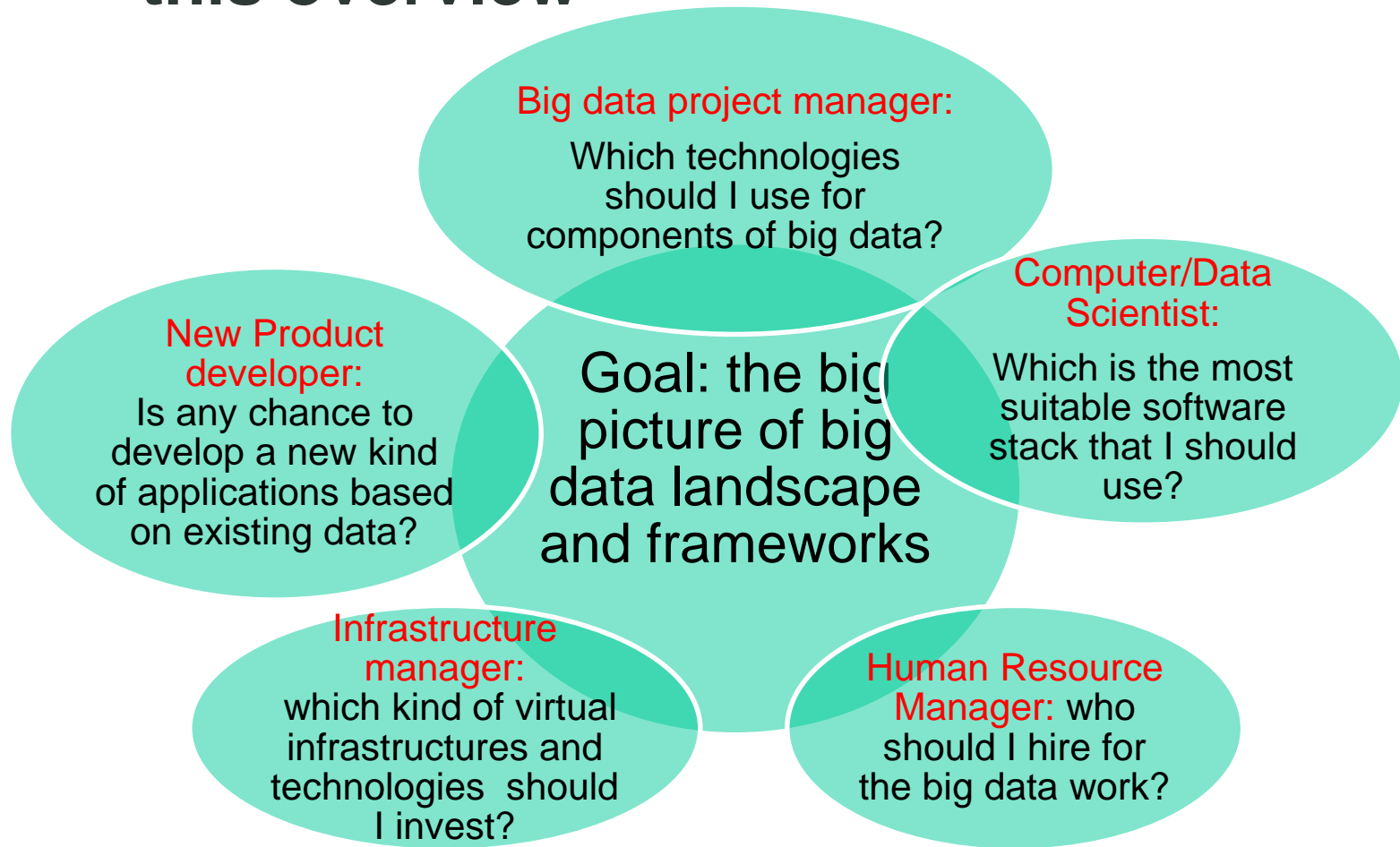


# Complex Data Analytics in the Cloud

Hong-Linh Truong  
Faculty of Informatics, TU Wien

[hong-linh.truong@tuwien.ac.at](mailto:hong-linh.truong@tuwien.ac.at)  
<http://www.tuwien.ac.at/staff/truong>

# Goals and expectation for this overview



- What does it mean big data?
- Why do you have to care?
- What do we need to do in order to build “big data solutions” and what are the key services engineering issues?
- Which techniques can we use for building elastic infrastructures, handling communications and messaging, managing data, and processing data?
- Put things together

Data: facts, responses, events, measurement, etc.

```
{"station_id":"1160629000","datapoint_id":122,"alarm_id":310,"event_time":"2016-09-17T02:05:54.000Z","isActive":false,"value":6,"valueThreshold":10}
```

## What does it mean “Big data”?

### NYC Taxi Data

The official [TLC trip record dataset](#) contains data for over 1.1 billion taxi trips from January 2009 through June 2015, covering both yellow and green taxis. Each individual trip record contains precise location coordinates for where the trip started and ended, timestamps for when the trip started and ended, plus a few other variables including fare amount, payment method, and distance traveled.

[Open Big Data](#) / Telecommunications - SMS, Call, Internet - MI

[Description](#) [Tabular Preview](#) [API](#) [Resources](#)

#### Schema

1. **Square Id:** the Id of the square that is part of the [Milano GRID](#); TYPE: numeric
2. **Time Interval:** the beginning of the time interval expressed as the number of millisecond elapsed from the Unix Epoch on January 1st, 1970 at UTC. The end of the time interval can be obtained by adding 600000 milliseconds (10 minutes) to this value. TYPE: numeric
3. **Country code:** the phone country code of a nation. Depending on the measured activity this value assumes different meanings that are explained later. TYPE: numeric
4. **SMS-in activity:** the activity in terms of received SMS inside the Square Id, during the Time interval and sent from the nation identified by the Country code. TYPE: numeric
5. **SMS-out activity:** the activity in terms of sent SMS inside the Square Id, during the Time interval and received by the nation identified by the Country code. TYPE: numeric
6. **Call-in activity:** the activity in terms of received calls inside the Square Id, during the Time interval and issued from the nation identified by the Country code. TYPE: numeric
7. **Call-out activity:** the activity in terms of issued calls inside the Square Id, during the Time interval and received by the nation identified by the Country code. TYPE: numeric
8. **Internet traffic activity:** the activity in terms of performed internet traffic inside the Square Id, during the Time interval and by the nation of the users performing the connection identified by the Country code. TYPE: numeric

# Is it big?

A banner with a dark blue background featuring a network of glowing blue lines and nodes, resembling a data network or constellation. The text is centered and reads:

**THE DATA OPPORTUNITY IS EXPLODING**  
**1,013,542,233,932,178,414,371 BYTES**  
ESTIMATED SIZE OF TODAY'S DATA  
HORTONWORKS SOLUTIONS ENABLE ORGANIZATIONS TO MAXIMIZE THE  
POWER OF OPEN SOURCE TO DELIVER ON THE PROMISE OF BIG DATA.  
[LEARN MORE](#)

Screenshot from <https://hortonworks.com/>

# Why do we have big data now?

- Social media
  - data generated by human activities in the Internet
  - Facebook, Twitter, Google+, etc.
- Internet of Things (IoT)/Machine-to-Machine (M2M)
  - data generated through monitoring devices and environments
  - data generated through machines
- Advanced sciences
  - data generated by advanced instruments
  - earth observation (e.g., Sentinel satellites)
- Personal information (e.g., healthcare)
- Open and transparent government
  - open government data
- Etc.

# Asset and Store Management

- Asset Management
  - Data about cars, homes, etc.
  - Monitoring + analytics about them
- In-store management
  - Movement of goods/products based smart labels
  - Movement of shoppers (e.g., based on shopping baskets or trolley movement)
  - Time spent at different good shelves/product sections and queues
- Cross-store management
  - Global inventory and data about customer buying behaviors

# Characterize big data

- Big data is often characterized by the concepts of V\*: Volume, Variety, Velocity, Veracity and Valence
  - Volume: size (big size, large-data set, massive of small data)
  - Variety: complexity (formats, types of data)
  - Velocity: speed (generating speed, data movement speed)
  - Veracity: quality is very different (bias, accuracy, etc.)
  - Valence: “chemical” relationships among different types of data (w.r.t data combination)



# Why do we need to care?

- Because of the values of data!

- Top-down: Data economy.

- More data → more evidence → more business success

- Bottom-up

- Optimizing what we have

- The Unreasonable Effectiveness of Data” (Alon Halevy, Peter Norvig, and Fernando Pereira) → with **more data**, the same algorithm performs much **more better**!

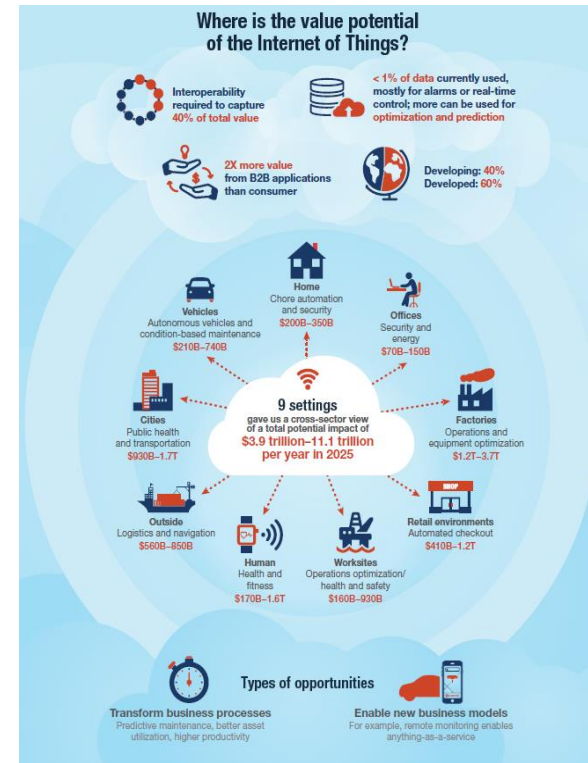


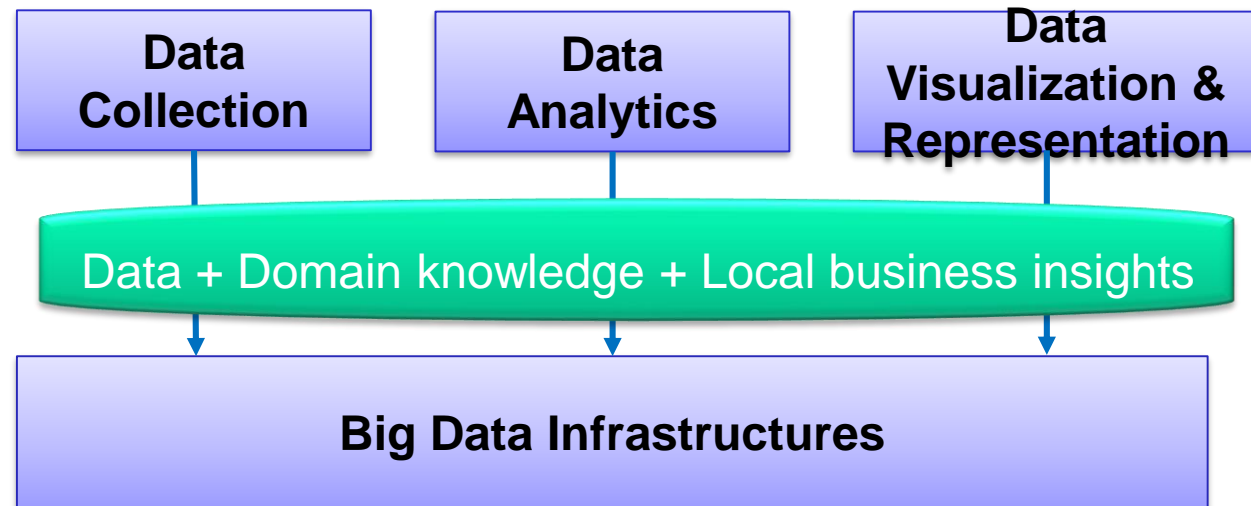
Figure source: McKinsey Global Institute: **THE INTERNET OF THINGS: MAPPING THE VALUE BEYOND THE HYPE** JUNE 2015 HIGHLIGHTS

Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The Unreasonable Effectiveness of Data. IEEE Intelligent Systems 24, 2 (March 2009). <http://static.googleusercontent.com/media/research.google.com/en/pubs/archive/35179.pdf>

# BIG DATA AND DATA SCIENCE

# Key components in big data

1. Engineering and operating software and data systems
2. Conducting data science tasks
3. Making sense of data analytics results



Data Science + Large-scale edge/cloud infrastructure + Service/System Engineering and Provisioning

## Key activities in data science

- Exploring and preparing data (business + technical people)
- Representing, modeling and transforming data (technical people)
- Analyzing data (technical people + domain expert + business people)
- Visualizing and presenting (resulting) data (technical/domain/business people)

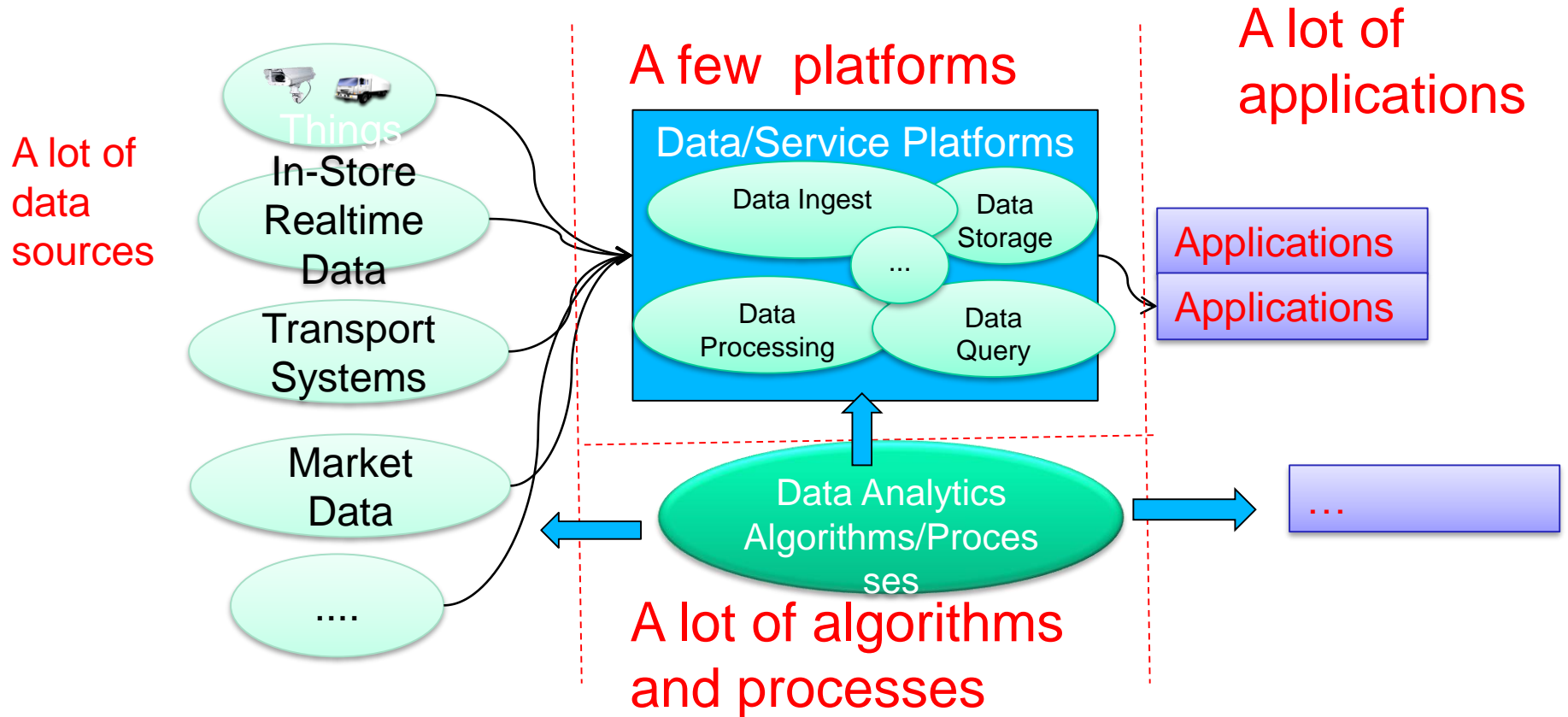
George Strawn, "Data Scientist", IT Professional, vol.18, no. 3, pp. 55-57, May-June 2016

So I can just learn databases and data mining?

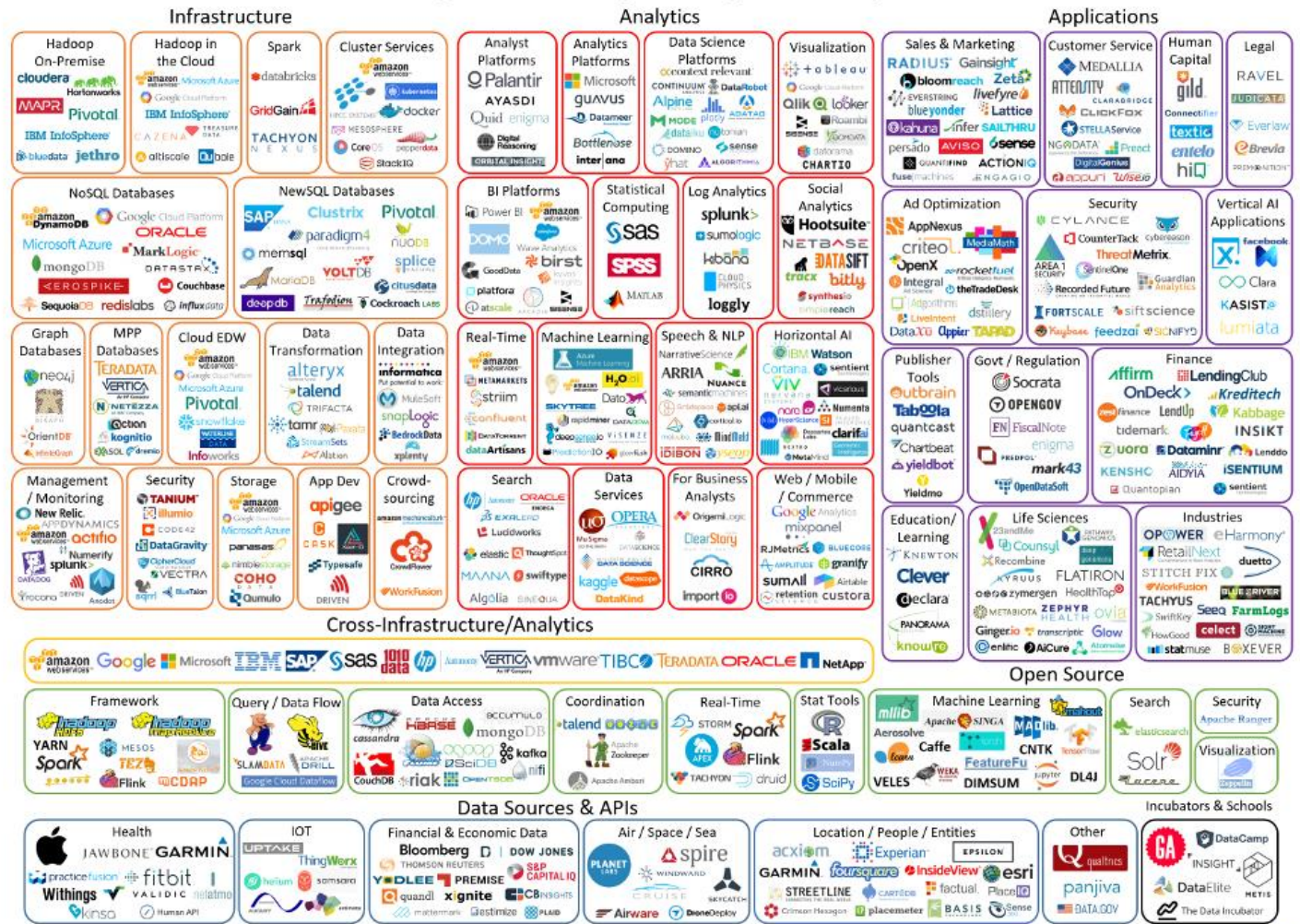
Wrong! you need others: computing systems, domain knowledge, etc

You need to work with an ecosystem of databases and tools

# Quantity aspects in big data



# Big Data Landscape 2016 (Version 2.0)



Last Updated 2/12/2016

© Matt Turck (@mattturck), Jim Hao (@jimrhao), & FirstMark Capital (@firstmarkcap)

FIRSTMARK

Source: [http://www.datameer.com/wp-content/uploads/2016/06/matt\\_turck\\_big\\_data\\_landscape\\_v11r.png](http://www.datameer.com/wp-content/uploads/2016/06/matt_turck_big_data_landscape_v11r.png)

**THINGS ARE TOO COMPLICATED!  
SO WHAT WOULD BE SUITABLE  
SERVICE ENGINEERING  
APPROACHES?**



# Big ETL: Extract, Transform and Load











- Traditional ETL: read data from databases (E), convert data, e.g., using rules (T), and write data to target databases (L)
- Big data ETL
  - **Big E**: read data from various sources, not just databases but also logs, instruments, middleware, etc., through various protocols
    - Different formats and data in motion and data at rest
  - **Big T**: various rules and services to perform transformations, complex processes for transforming data, batch and realtime data transformation
  - **Big L**: various types of target sinks (databases, middleware, services, etc.) using various protocols



# Interaction: protocols & interfaces

- Large number of communication protocols and interfaces
- Interaction styles, protocols and interfaces
  - REST, RPC, Message Passing, Stream-oriented Communication, Distributed Object models, Component-based Models
  - Your own protocols
- Other criteria
  - Architectural constraints
  - Scalability, Performance, Adaptability, Monitoring, Logging, etc.

# Polyglot programming for big data

Language Rank	Types	Spectrum Ranking
1. C		100.0
2. Java		98.1
3. Python		98.0
4. C++		95.9
5. R		87.9
6. C#		86.7
7. PHP		82.8
8. JavaScript		82.2
9. Ruby		74.5
10. Go		71.9

Source:

<http://spectrum.ieee.org/computing/software/the-2016-top-programming-languages>

- Polyglot programming is important for big data
  - We need multiple programming languages
- In big data software ecosystems
  - Java
  - Python
  - R
  - Scala
  - NodeJS
- Well support by industries and powerful open source frameworks

# Python for data science/big data analytics

- [www.python.org](http://www.python.org)
- Most big data platforms support Python APIs for accessing services and infrastructures
- Several data analytics can be written in python: Natural Language Processing (e.g., for sentiment analytics), Recommendation (e.g., buy a product)
- Several libraries for python
  - NumPy – numeric python <http://www.numpy.org/>
  - StatsModels <http://statsmodels.sourceforge.net/>
  - Scikit <http://scikit-learn.org/stable/>
  - ML <http://mlpy.sourceforge.net/>
  - Python Data Analysis Library: <http://pandas.pydata.org/>
- Many machine learning/Deep learning
  - Tensor (google): <https://www.tensorflow.org/>
  - Theano: <http://deeplearning.net/software/theano/>
- Workflow in Python (e.g., Airflow)

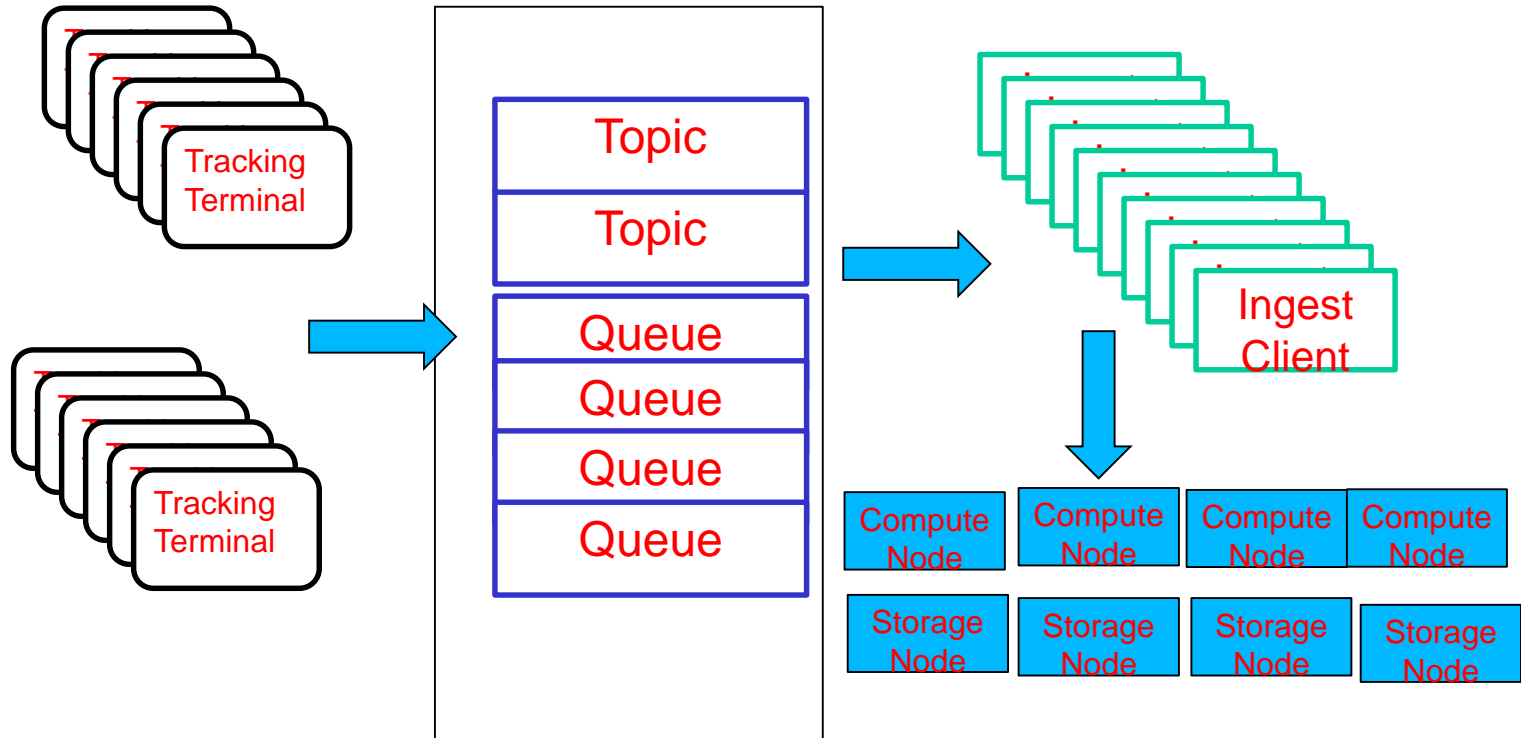
 Microsoft | Developer

Python Engineering at Microsoft  
Using Python to drive innovation

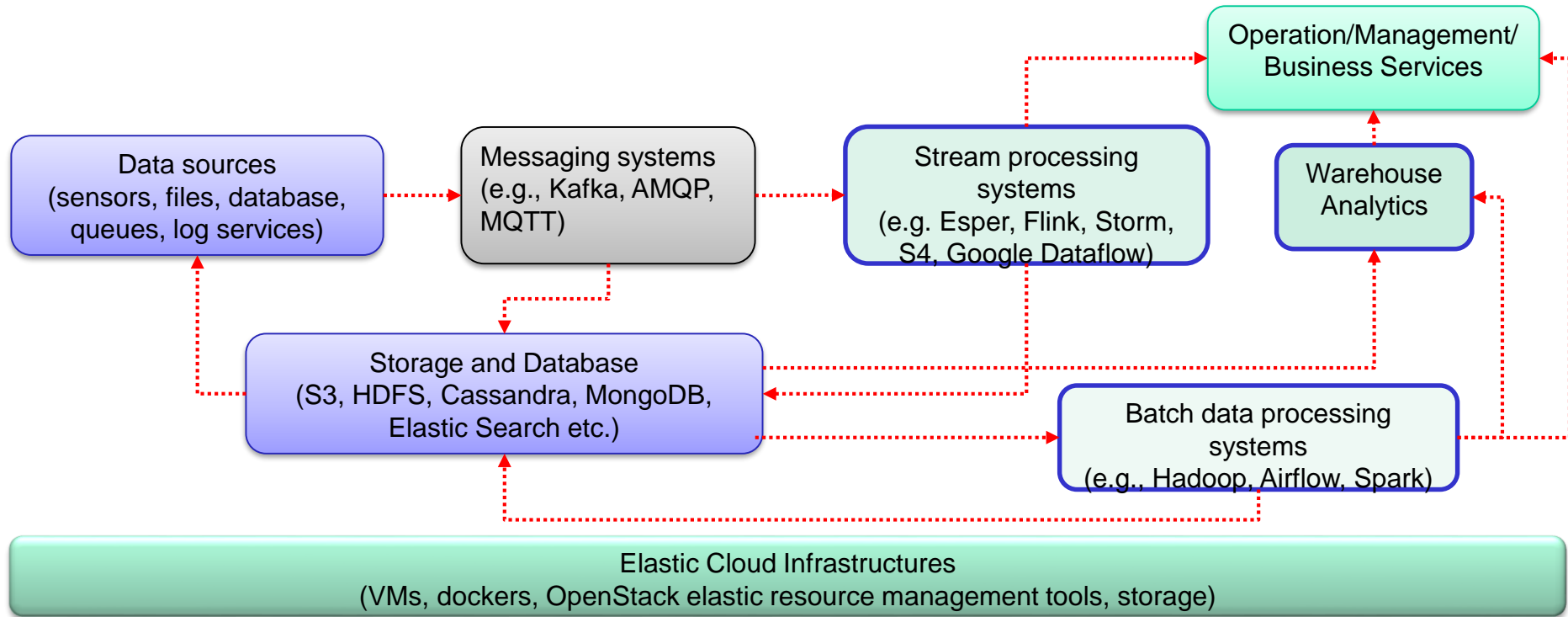
# Concurrent and Parallel Programming models

- Why are they important?
  - Faster, faster and faster!
- Embarrassingly parallel workload/pattern is one of the most popular ones for big data analytics
  - Independent parallel data processing jobs
- Implementation
  - Programming languages + programming models + job management
- Examples of models and tools
  - MapReduce, Dataflow/Workflow, MPI, AKKA

# The big problem



# Big data at large-scale



# WHICH KIND OF INFRASTRUCTURES DO I NEED?

# Understanding resource management

- Resources:
  - Provide data, computing, and network function capabilities
  - Within a data center and across multiple data centers
- Tasks/Jobs:
  - Functions: transferring data, monitoring collecting
  - Lifetime: short versus long
  - Mode: batch versus continuous running
  - System or application jobs
- Big data analytics require multiple types of resources for running diverse types of jobs
- Big infrastructures for big data analytics need to manage jobs and resources



# **TO BUILD BIG INFRASTRUCTURES YOU NEED VIRTUALIZATION AND ELASTICITY**

# HOW CAN I DEAL WITH A LOT OF MESSAGES ?

# Some key issues

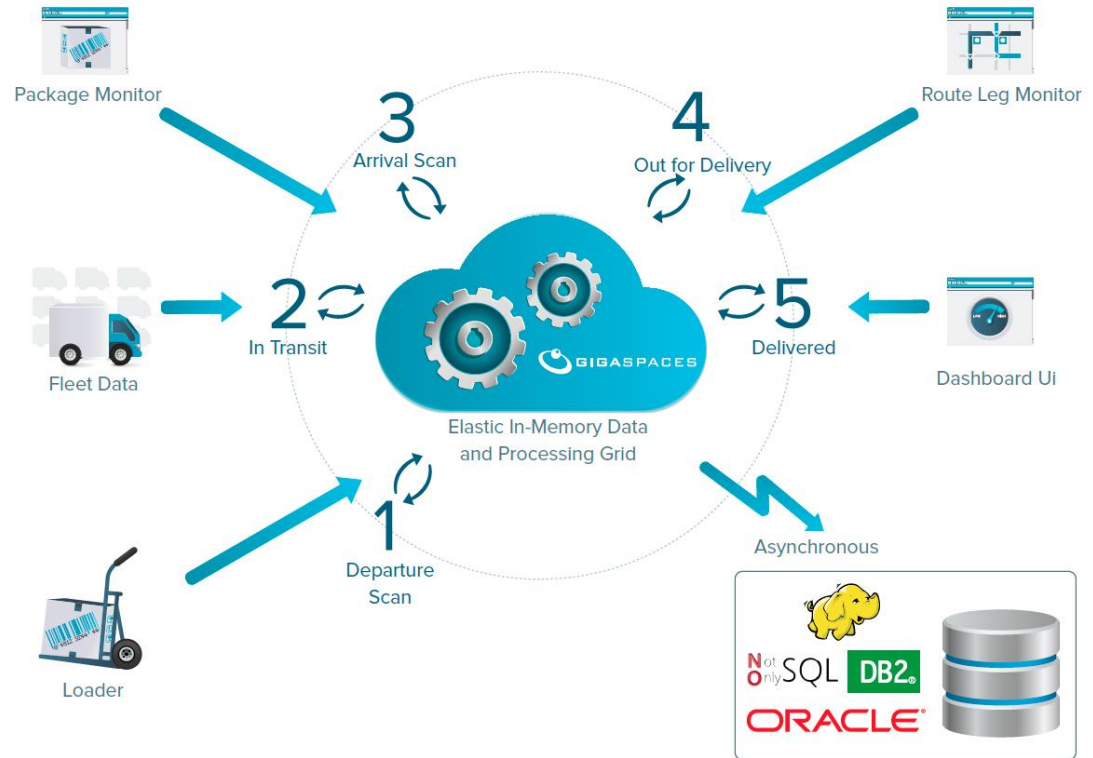
- Data exchange:
  - a lot of streaming data (events/monitoring data), many log files to be handled, short messages (but a lot) for controls and notifications
- Transport protocols & Message format/syntax and message semantics (applications/systems specific)
- Places: within infrastructures and between the producer and the consumer
- Some important guarantees
  - Reliability: message loss and duplication
  - Interoperability: interoperable format and processing
  - Performance: high processing rates, low latency, etc.

# WHICH DATA MODELS SHOULD I USE?

- A lake of data
    - Ingest and integrate as many as possible types of data
    - To archive a lot of data so that potentially many analytics and applications can access
- Data lake is a concept so you can implement it based on your requirements and needs

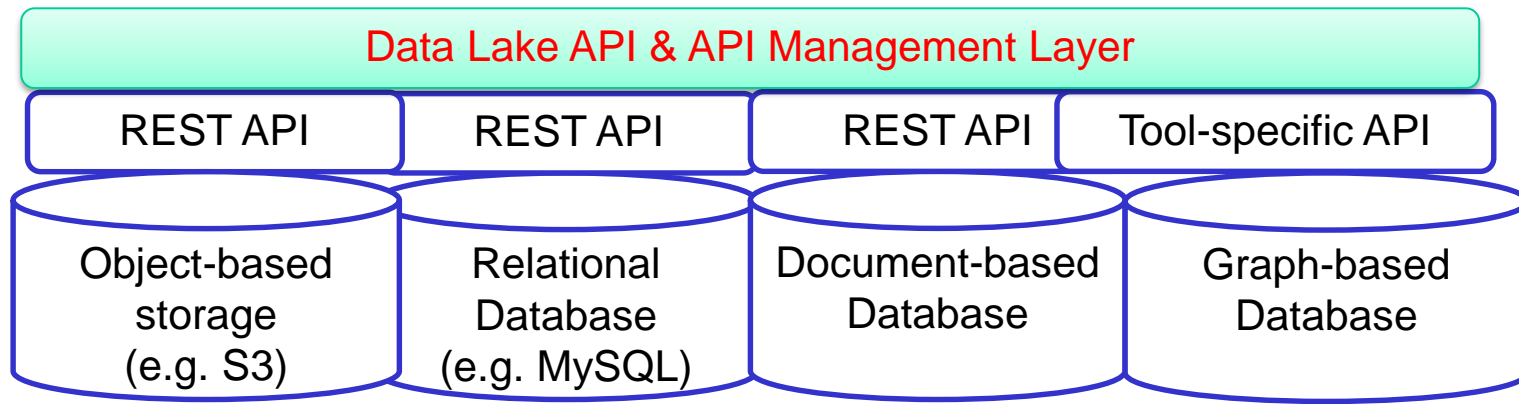
# Example

Can we build a data lake using the concept of “data space”



Source: <http://www.gigaspace.com/logistics-and-shipping-management>

# Data Lake through Data Access API & API Management

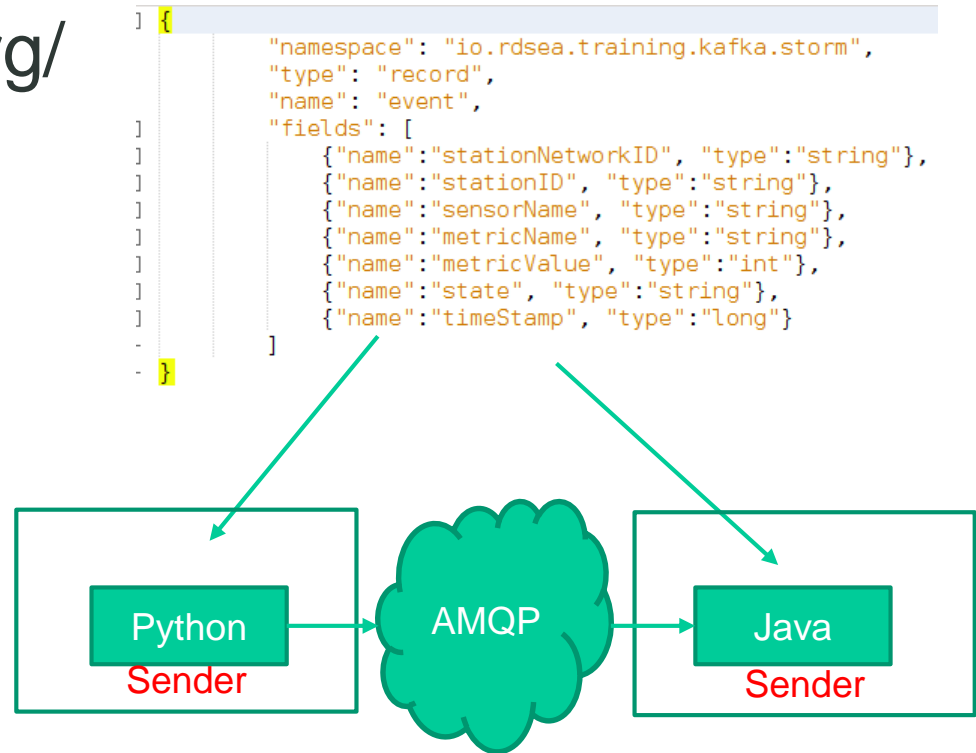


- Data access APIs can be built based on well-defined interfaces
- Help to bring the data object close to the programming language objects

**BUT HOW CAN I INGEST A LOT  
OF DATA INTO MY BIG DATA  
INFRASTRUCTURES?**



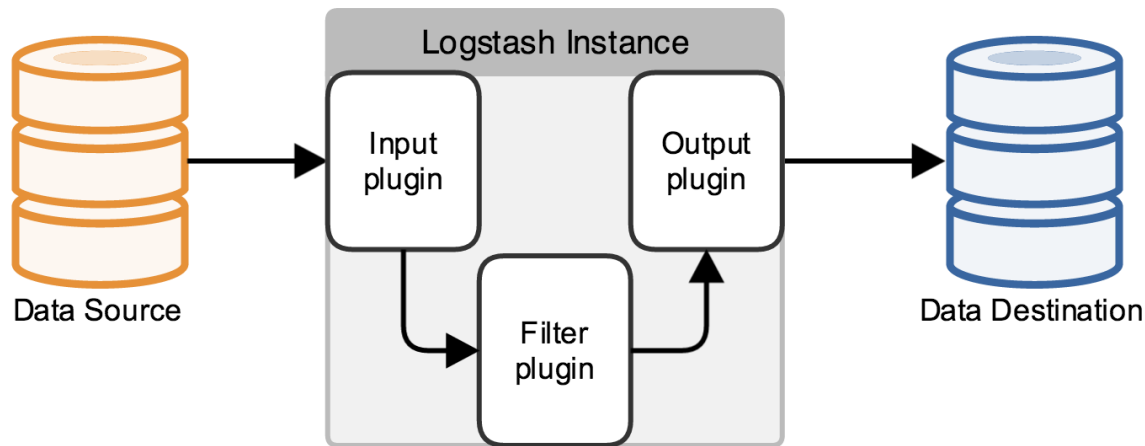
- <https://avro.apache.org/>
- Support message description
- Serialize and deserialize libraries
- Work with different languages



# Some other techniques

- Protobuf
  - From Google
  - <https://github.com/google/protobuf>
  - Language-neutral, platform-neutral mechanism for serializing/deserializing structured data
- Thrift
  - <https://thrift.apache.org>
  - Support also serializing and deserializing data)
  - Support cross-language services development
    - Specify services interfaces
    - Data exchange
    - Code generation

# Logstash

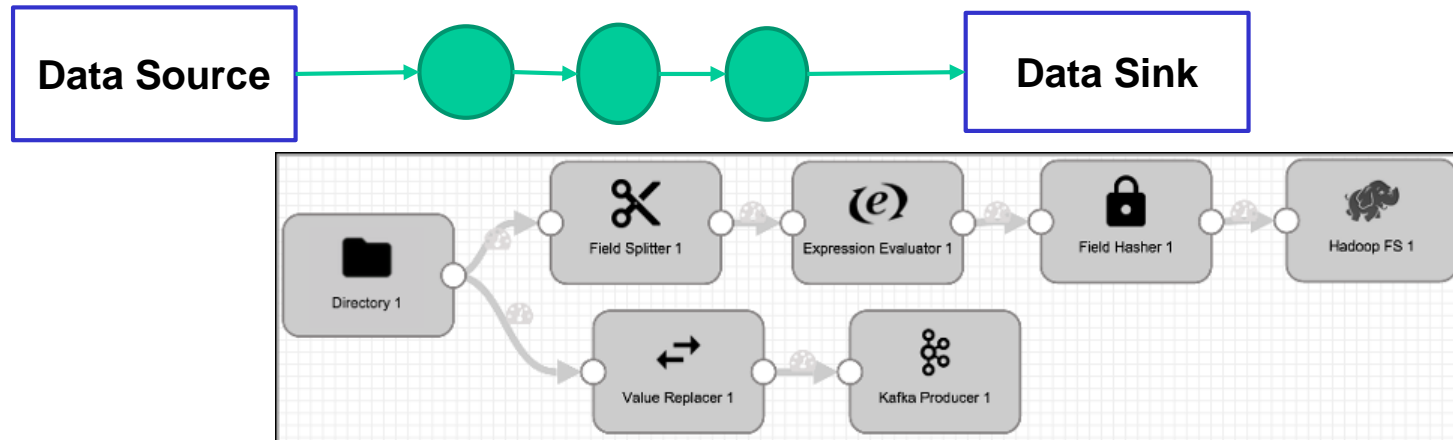


- Codecs: stream filters within inputs or outputs that change data representation
- E.g.: multilines → a single event

Source: <https://www.elastic.co/guide/en/logstash/current/advanced-pipeline.html>

# Streamsets

- <https://streamsets.com>
- Using data pipeline to filter, aggreage, etc.
- Apply to data in motion
  - Support „at least once and „at most once“ data delivery gurantees
- Interface to various data sources and sinks (Cassandra, Hadoop, Elastic Search, ...)

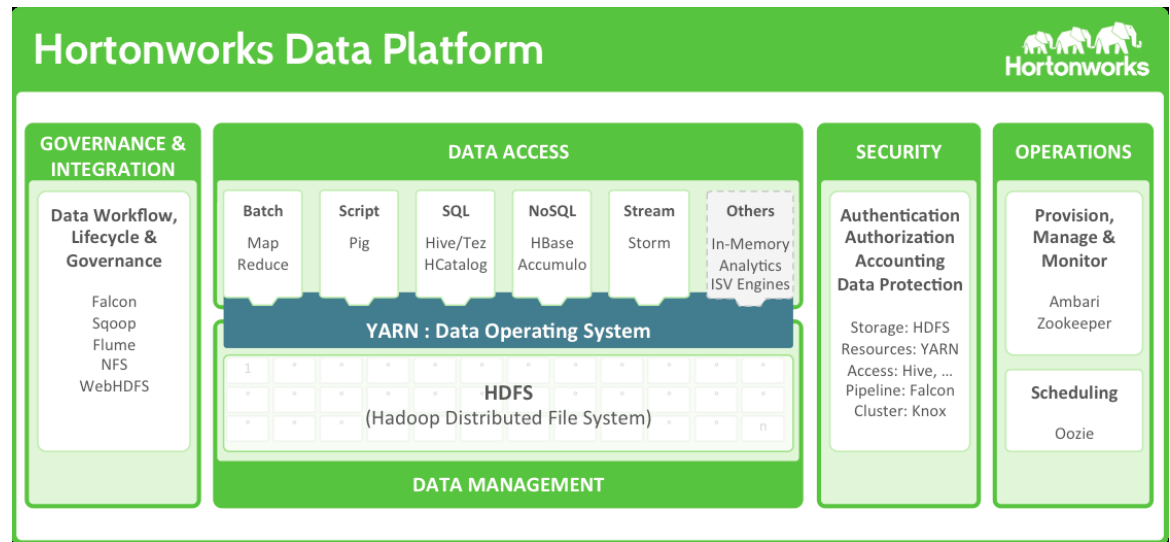


Source:  
<https://streamsets.com>

# HOW CAN I DO DATA PROCESSING?

# Hadoop ecosystem

- Built around Mapreduce programming models and Hadoop software ecosystems
  - <http://hadoop.apache.org/>
- From “The Forrester Wave™: Big Data Hadoop Distributions, Q1 2016”: Top Hadoop solution providers are **Cloudera, Hortonworks, IBM, MapR Technologies, and Pivotal Software**



Source: <http://hortonworks.com/blog/defining-enterprise-hadoop/>

# Spark ecosystem

Programming with Java, Scala, Python, R  
We can have a separate modules

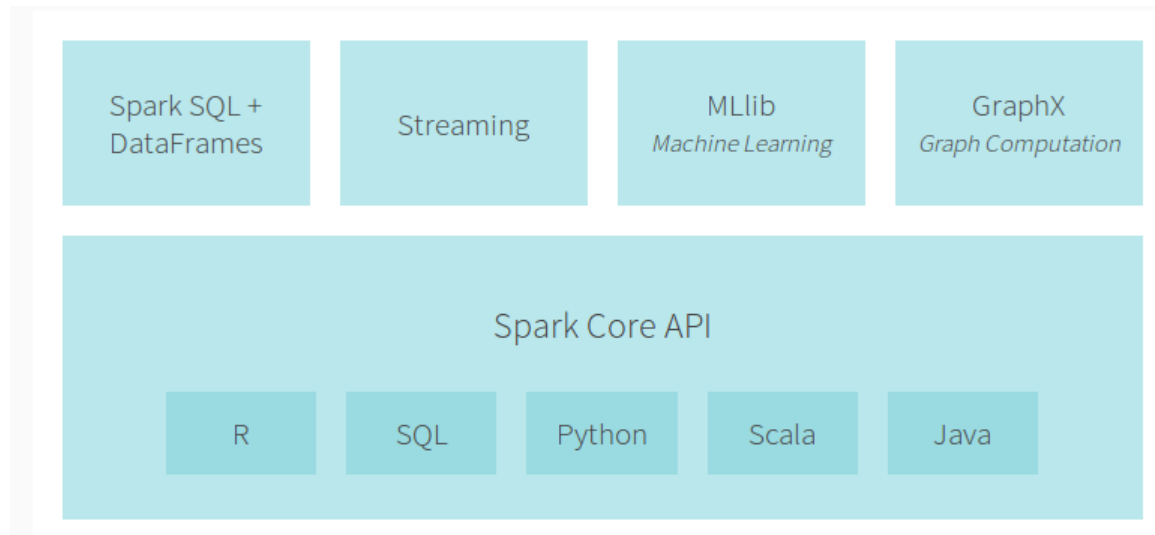
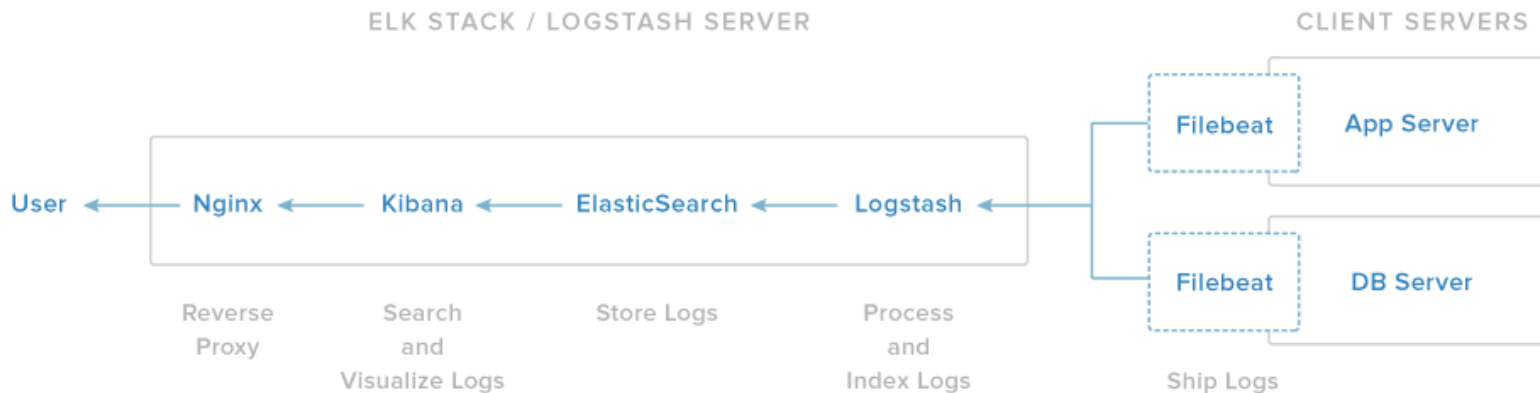


Figure source:  
<https://databricks.com/spark/about>

# ELK Stack

- Building using elastic components: Elasticsearch, Elasticsearch Hadoop, Kibana, and Logstash
- <https://www.elastic.co/>



Source: <https://www.digitalocean.com/community/tutorials/how-to-install-elasticsearch-logstash-and-kibana-elk-stack-on-ubuntu-14-04>



- Main from services of Influx
  - <https://www.influxdata.com>
- Focus on time series data
- The same principles
  - Collect
  - Storage
  - Visualize
  - ETL

## TICK

### Telegraf

- Time-Series Data Collector

### InfluxDB

- Time-Series Data Storage

### Chronograf

- Time-Series Data Visualization

### Kapacitor

- Time-Series Data Processing

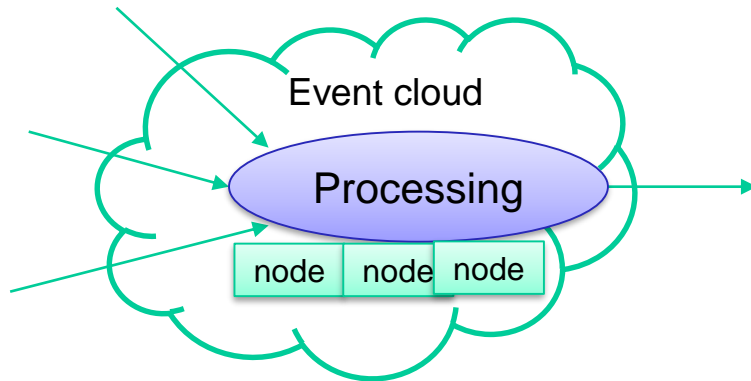
# Pipeline/Workflow

- We often need complex workflow for data analytics
  - Invoke different activities executed by different frameworks/services
- E.g.
  - Netflix: <https://medium.com/netflix-techblog/meson-workflow-orchestration-for-netflix-recommendations-fc932625c1d9>
  - AirBnB: <https://medium.com/airbnb-engineering/airflow-a-workflow-management-platform-46318b977fd8>
  - Facebook: <https://research.fb.com/publications/sve-distributed-video-processing-at-facebook-scale/>
  - Microsoft ML: <https://studio.azureml.net/>
  - Google: <https://cloudacademy.com/blog/google-prediction-api/>

# Centralized versus distributed processing topology

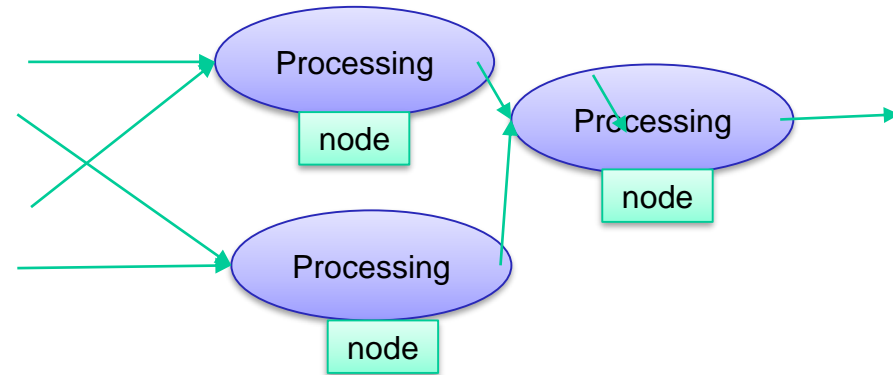
Two views: **streams of events** or **cloud of events**

Complex Event Processing  
(centralized processing)



Usually only  
queries/patterns are  
written

Streaming Data Processing  
(distributed processing)



Code processing events and  
topologies need to be  
written

- Apache Storm
- Apache Spark
  - Streaming processing but using micro-batching
- Apache Apex
- Apache Flink
- Apache Kafka Streaming & Streaming SQL
- etc.

# Your next assignment

1. Choose a big data analytics problem/application domain
  - (Industry 4.0/Cloud Manufacturing, IoT for Retail, IoT for Healthcare)
2. Discuss software ecosystem and pipelines
  - Processing frameworks, infrastructures
  - From collection to analytics pipelines
  - Machine learning pipelines
3. Demonstrate with small design and implementation
  - How easy to glue data, services, processes, etc. together
  - Coupling services for big data analytics
  - Algorithms + frameworks

# Thanks for your attention

Hong-Linh Truong  
Faculty of Informatics, TU Wien  
[hong-linh.truong@tuwien.ac.at](mailto:hong-linh.truong@tuwien.ac.at)  
<http://www.infosys.tuwien.ac.at/staff/truong>