How to build and run a big data platform in the 21st century

Ali Dasdan - Atlassian - adasdan@atlassian.com
Dhruba Borthakur - Rockset - dhruba@rockset.com

This tutorial was presented at the IEEE BigData Conference in Los Angeles, CA, USA on Dec 9th, 2019.

Disclaimers

- This tutorial presents the opinions of the authors. It does not necessarily reflect the views of our employers.
- This tutorial presents content that is already available in the public domain. It does not contain any confidential information.

Speaker: Ali Dasdan

Ali Dasdan is the head of engineering for Confluence Cloud at Atlassian. Prior to Atlassian, he worked as the head of engineering and CTO in three startups (Turn in real-time online advertising, Vida Health in chronic disease management, Poynt in payments platform) and as an engineering leader in four leading tech companies (Synopsys for electronic design automation; Yahoo! for web search and big data; Ebay for e-commerce search, discovery, and recommendation; and Tesco for big data for retail). He has been working in big data and large distributed systems since 2006 (releasing the first production application on Hadoop in 2006). He led the building of three big data platforms on-prem from scratch (100PB+ capacity at Turn, multi-PB capacity for a leading telco, and 30PB+ capacity at Tesco). He is active in multiple areas of research related to big data, namely, big data platforms, distributed systems, machine learning. He received his PhD in Computer Science from University of Illinois at Urbana-Champaign in 1999. During part of his PhD, he worked as a visiting scholar at the University of California, Irvine on embedded real-time systems.

Linkedin: https://www.linkedin.com/in/dasdan/

Scholarly publications: https://dblp.org/pers/hd/d/Dasdan:Ali

Citations: https://scholar.google.com/citations?user=Bol1-tMAAAAJ&hl=en

Courses taught: Multiple courses at the graduate school, many paper presentations at conferences, 3-day training on embedded real-time systems at a SoCal company, two tutorials in the 18th and 19th International World Wide Web Conferences.

Speaker: Dhruba Borthakur

Dhruba Borthakur is cofounder and CTO at Rockset (https://rockset.com/, a company building software to enable data-powered applications. Dhruba was the founding engineer of the open source RocksDB [1] database when we was an engineer at Facebook. Prior to that, he is one of the founding engineers of the Hadoop File System [2] at Yahoo!. Dhruba was also an early contributor to the open source Apache HBase project. Previously, he was a senior engineer at Veritas Software, where he was responsible for the development of VxFS and Veritas SanPointDirect storage system; was the cofounder of Oreceipt.com, an ecommerce startup based in Sunnyvale; and was a senior engineer at IBM-Transarc Labs, where he contributed to the development of Andrew File System (AFS). Dhruba holds an MS in computer science from the University of Wisconsin-Madison.

Linkedin: https://www.linkedin.com/in/dhruba/

Scholarly publications: https://dblp.uni-trier.de/pers/hd/b/Borthakur:Dhruba

Citations: https://scholar.google.com/citations?user=NYgWyv0AAAAJ&hl=en.

Rockset: : https://rockset.com/

RocksDB: https://en.wikipedia.org/wiki/RocksDB

Apache HDFS: https://svn.apache.org/repos/asf/hadoop/common/tags/release-0.10.0/docs/hdfs_design.pdf

Abstract

We want to show that building and running a big data platform for both streaming and bulk data processing for all kinds of applications involving analytics, data science, reporting, and the like in today's world can be as easy as following a checklist. We live in a fortunate time that many of the components needed are already available in the open source or as a service from commercial vendors. We show how to put these components together in multiple sophistication levels to cover the spectrum from a basic reporting need to a full fledged operation across geographically distributed regions with business continuity measures in place. We plan to provide enough information to the audience that this tutorial can also serve as a high-level goto reference in the actual process of building and running.

Motivation

We have been working on big data platforms for more than a decade now. Building and running big data platforms have become relatively more manageable now. Yet our interactions at multiple forums have shown us that a comprehensive and checklist-style guideline for building and running big data platforms does not exist in a practical form. With this tutorial we are hoping to close this gap, within the limitations of the tutorial length.

Targeted audience

- Targets
  - Our first target is the developers and practitioners of software or analytics or data science. However, we do provide multiple research issues and provide a review of the research literature so that researchers and students can significantly benefit from this tutorial too.
- Content level
  - 25% beginner, 50% intermediate, 25% advanced
- Audience prerequisites
  - Introductory software engineering or data science knowledge or experience as an individual contributor or as a manager of such individuals. Interest in building and running big data platforms.

Terminology and notation

- **API**: Application Programming Interface
- **BCP**: Business Continuity Planning
- **DAG**: Directed Acyclic Graph
- **DB**: Database
- **DR**: Disaster Recovery
- **GDPR**: General Data Protection Regulation
- **ML**: Machine Learning
- **On premise (on-prem)**: Computing services offered by the 1st party
- **Public cloud**: Computing services offered by a 3rd party over the public internet
  - Possibly from Alibaba, Amazon, Google, IBM, Microsoft, Oracle, etc.
- **RPO:** Recovery Point Objective; **RTO:** Recovery Time Objective
- **SLA**: Service Level Agreement; **SLI**: Service Level Indicator; **SLO**: Service Level Objective
- **TB** (terabyte); **PB** (petabyte) = 1000 TB; **EB** (exabyte) = 1000 PB

Tutorial outline

	Section	Time allocated	Presenter
1	Introduction	15	Ali
2	Architecture	30	Ali
3	Input layer	10	Dhruba
4	Storage layer	10	Dhruba
5	Data transformation	10	Ali
6	Compute layer	10	Ali
7	Access layer	10	Dhruba
8	Output layer	10	Dhruba
9	Management and operations	10	Ali
10	Examples, lessons learned	5	Ali

Introduction

Section 1 of 10

What is big data?

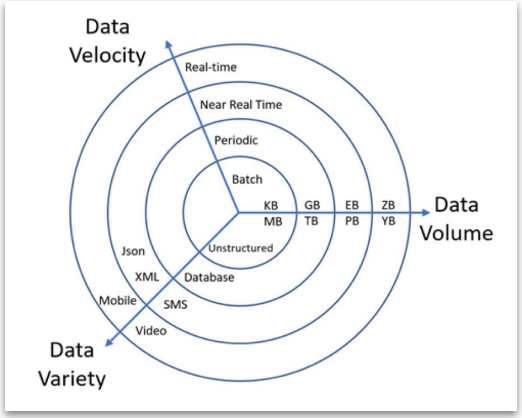
- Clive Humby, UK mathematician, founder of Dunnhumby, and an architect of Tesco's Clubcard, 2006
  - "Data is the new oil. It's valuable, but if unrefined it cannot really be used. It has to be changed into gas, plastic, chemicals, etc to create a valuable entity that drives profitable activity; so must data be broken down, analyzed for it to have value."

https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data

- The Economist (2019): "Data is the new oil."
- Forbes (2018), Wired (2019): "No, data is not the new oil."
- Ad Exchanger (2019): "Consent, not data, is the new oil."

What is big data?

- No standard definition
- According to my then 6-yr old daughter
  - “You get data; then you get more data; and you get more and more data; finally you get big data.”
- Another definition
  - If the size of your data is part of the problem, you may have big data.
- “Big” along one or more of 3 V’s



Data volumes

- 10+ of terabytes
  - A high end server
- 10+ of petabytes
  - US Library of Congress (10PB?)
- 100+ of petabytes
  - Turn (now Amobee) (2016)
  - Dropbox (2016)
  - Netflix (2017)
  - Twitter (2017)
- 1+ of exabytes
  - Amazon, Google, Facebook
  - 1 exabyte = 1000 petabytes
- 1+ of zettabytes
  - (per Cisco) global internet traffic

How big is a petabyte of data?

How many?	What?	Size in bytes (roughly)
1	Character	1 byte (B)
1	Page of text (~2000 characters)	2 kilobytes (KB)
1	File cabinet drawer (~2500 pages)	5 megabytes (MB)
1	4-drawer file cabinet	20 megabytes
50	4-drawer file cabinets	1 gigabyte (GB)
50,000	4-drawer file cabinets	1 terabyte (TB)
50,000,000	4-drawer file cabinets	1 petabyte (PB)

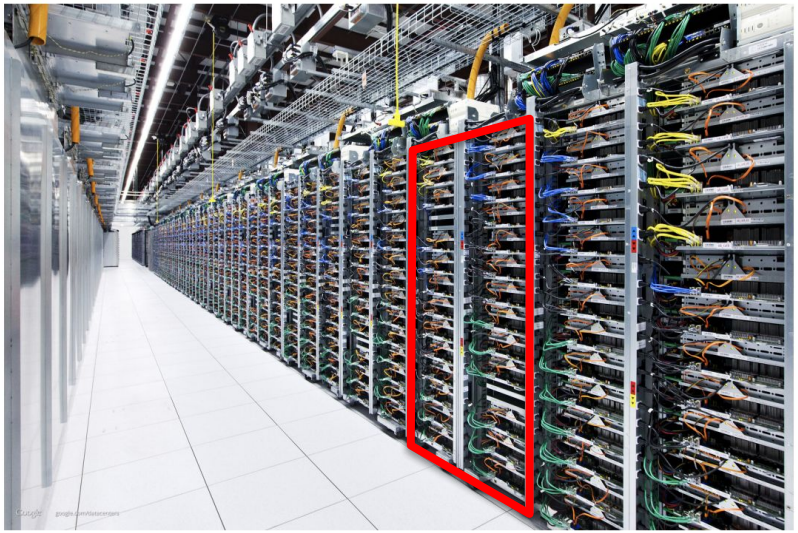
How big is a petabyte of data?

- Length of US land and water boundary: 19,937mi = 24,483km
  - Length of US-Canada land and water boundary: 5,525mi = 8,892km
  - Length of US-Mexico land and water boundary: 1,933mi = 3,112km
  - Length of US coastline: 12,479mi = 20,083km
- Recall 1 PB of text = 50M file cabinets or 23,000km of file cabinets
- So 1PB of text almost covers all around US land and water boundary

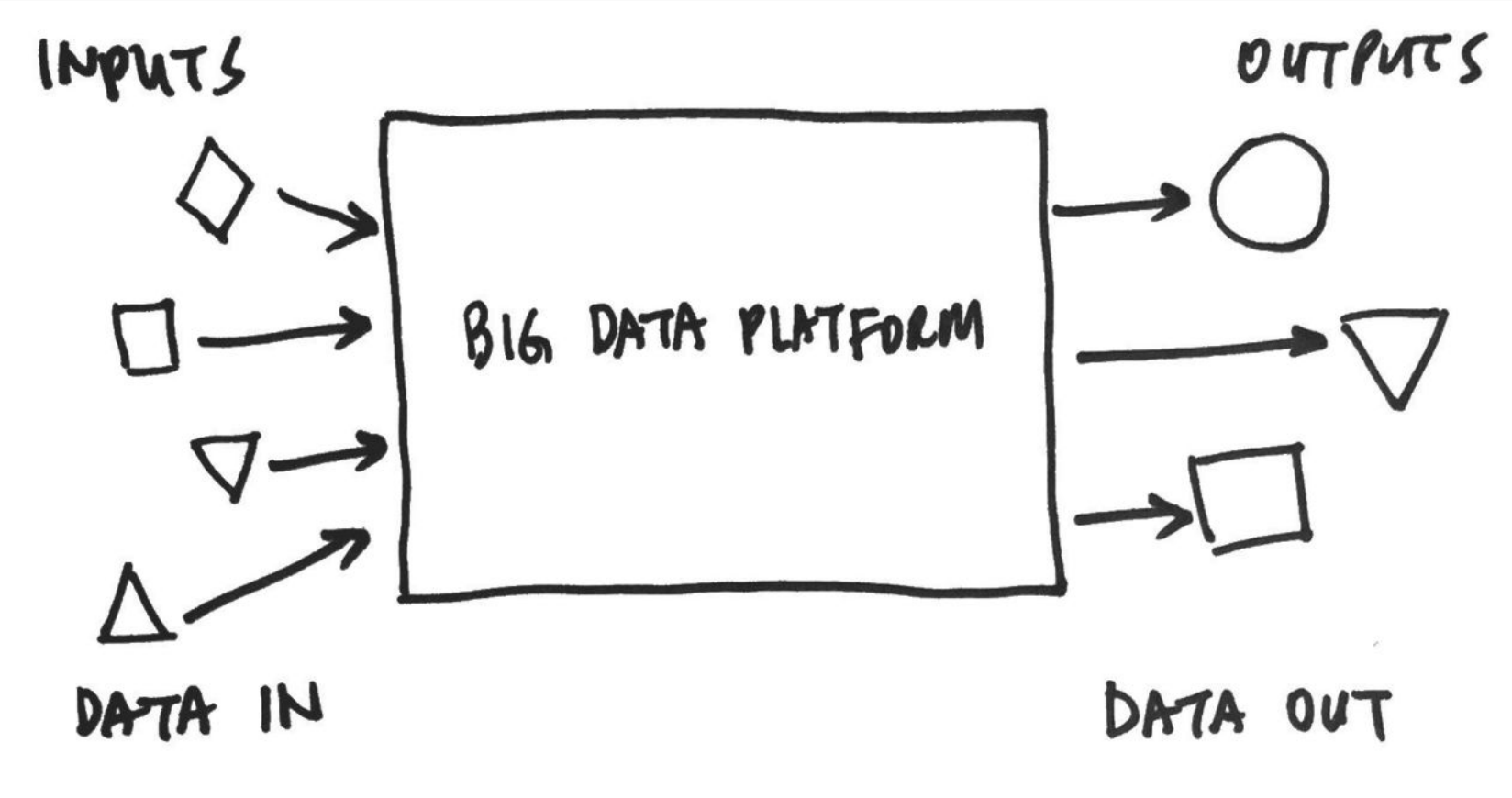
- https://fas.org/sgp/crs/misc/RS21729.pdf

Where to fit a petabyte of data in a datacenter?

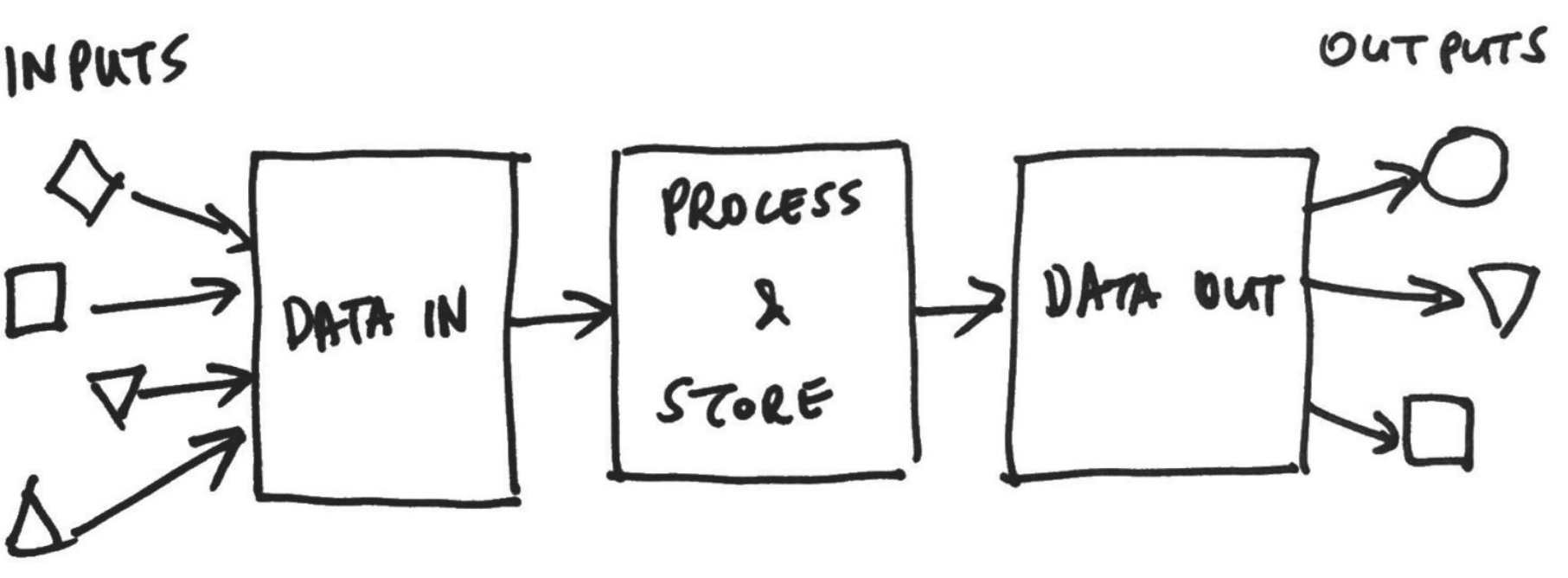
- A high-end server in 2017
  - Main memory size: 0.5TB
  - Hard disk size: 45TB
  - Number of cores: 40
- 2 racks with 15 servers each
  - Total main memory size: 15TB
  - Total hard disk size: 1.4PB
  - Total number of cores: 1,200
- Now consider the total number of racks in the row in this picture
- Then consider the total number of rows in a typical datacenter
- Finally consider the total number of data centers owned by one or more companies
- That is a lot of data capacity



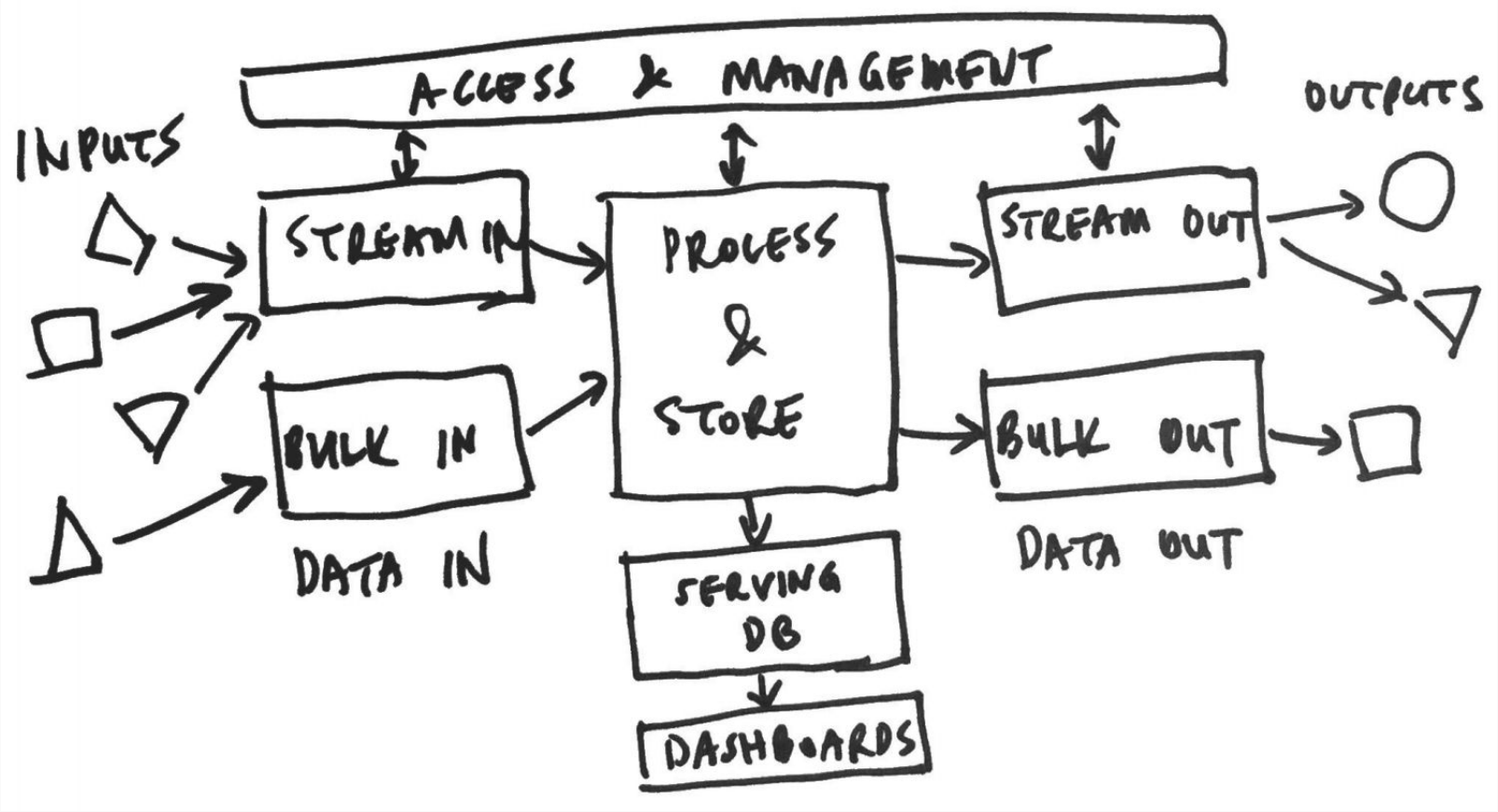
Generic big data platform architecture (1 of 3)



Generic big data platform architecture (2 of 3)

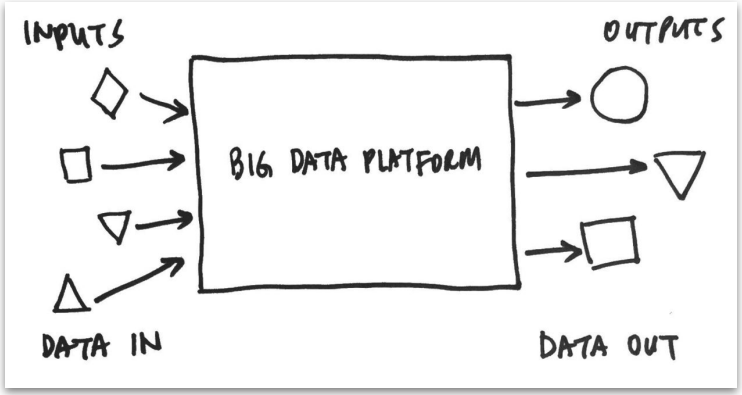


Generic big data platform architecture (3 of 3)



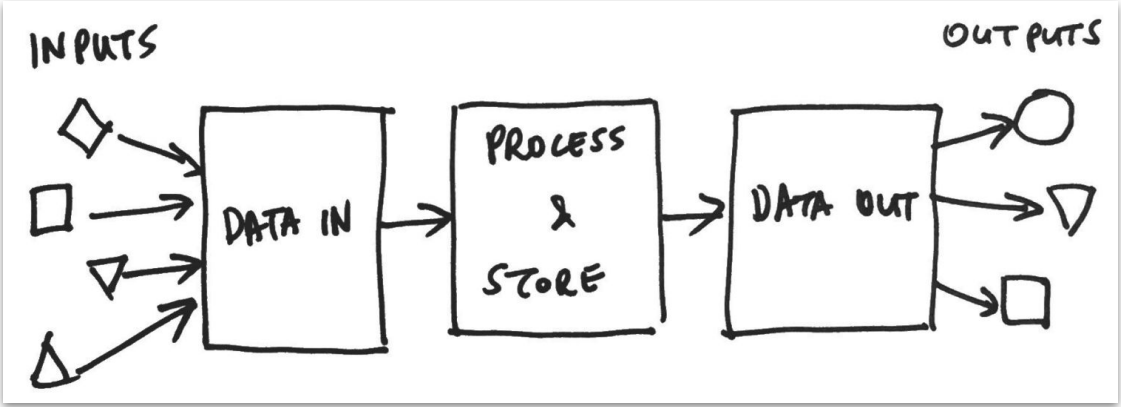
Generic big data platform

- To discuss
  - how this generic architecture is realized in the real world, or
  - how the real world architectures fit into this generic architecture
- To discuss the six key components of a big data platform

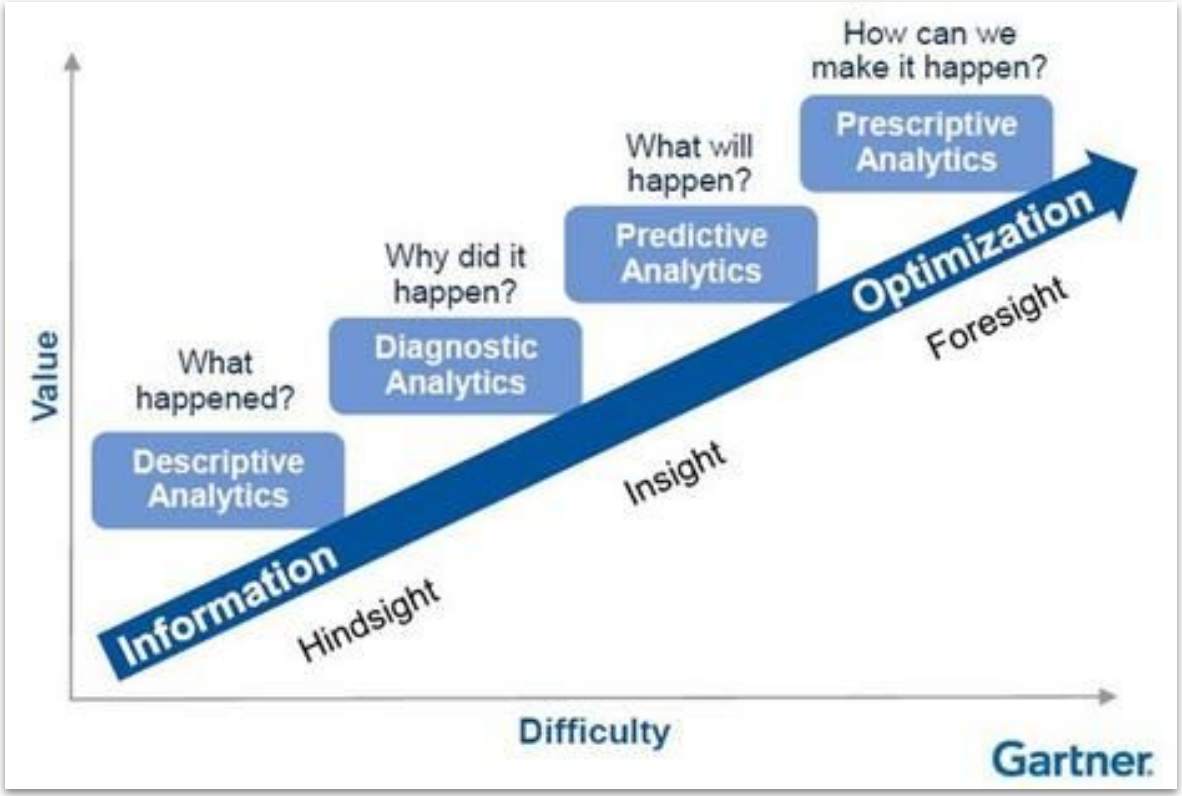


Six key components to discuss

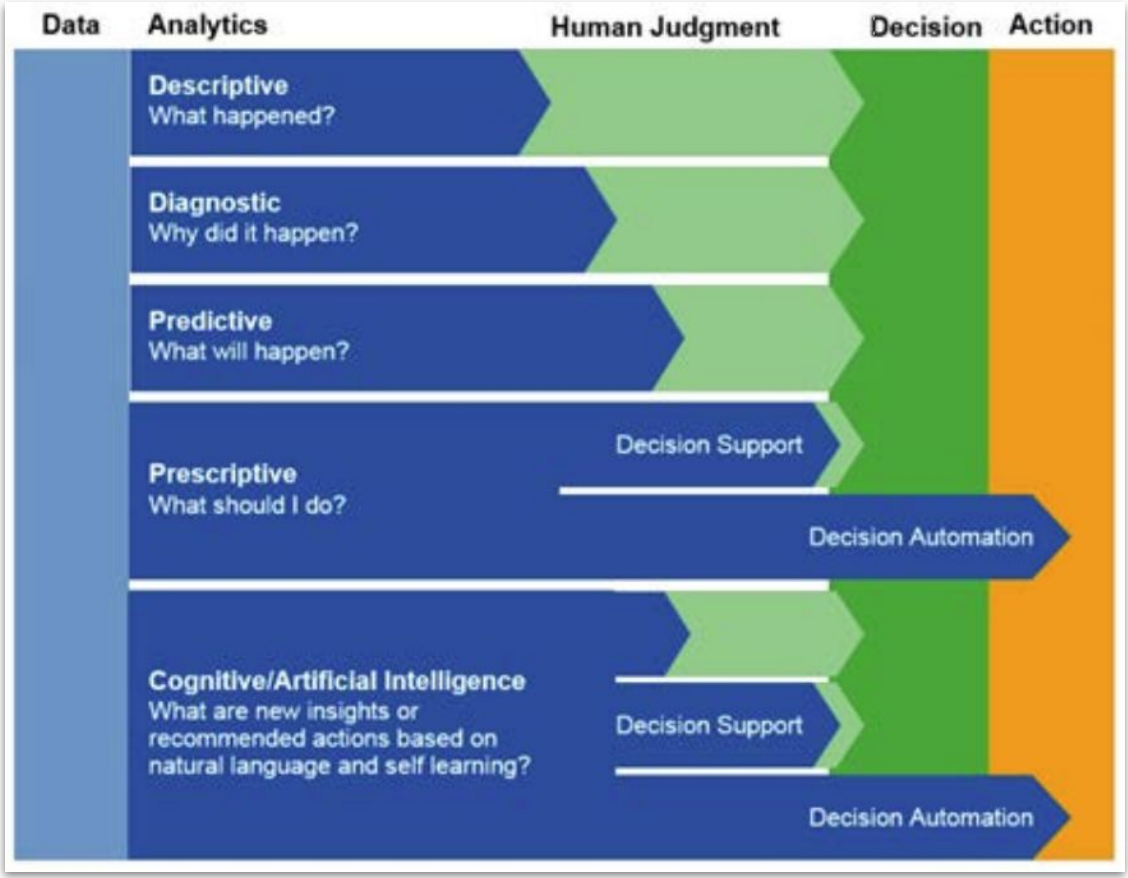
1. Input
2. Compute (or process)
3. Data transformations
4. Storage
5. Access
6. Output
7. Management and operations



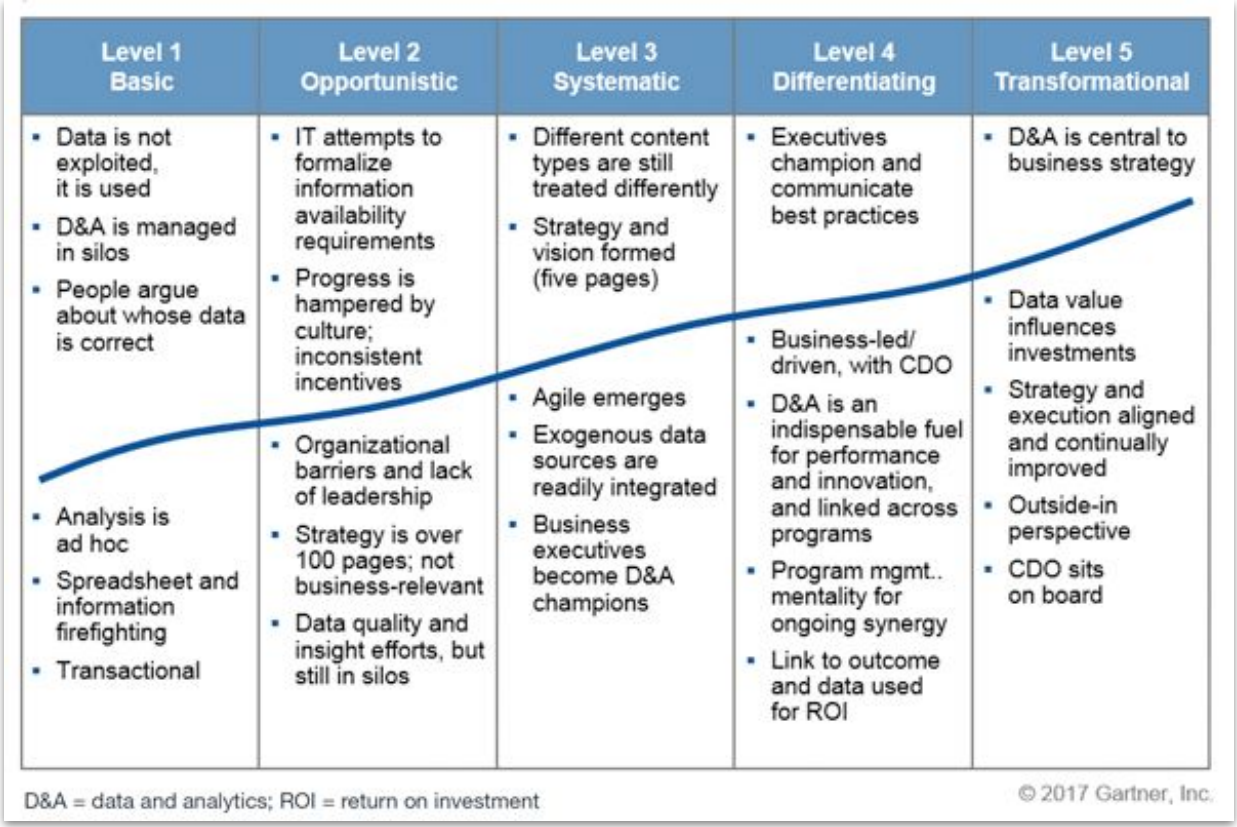
Gartner: Analytics maturity (2017)



Gartner: Types of analytics (2017)



Gartner: Analytics maturity (2017)



https://www.gartner.com/en/newsroom/press-releases/2018-02-05-gartner-survey-shows-organizations-are-slow-to-advance-in-data-and-analytics

Cons of a big data platform

- Hope that pros are obvious already
- Cons and how to counter them
  - Costly
    - But the benefits are immense, just a little more patience
  - Not many users
    - But they will come, guaranteed
  - Centralized
    - But this opens up more innovation opportunities through sometimes surprising connections between data sources and applications
  - Single point of failure
    - But the platform can be made more reliable using disaster recovery and other availability and reliability enhancement techniques
  - Big security vulnerability
    - But the platform can be made more secure using many security enhancement techniques

Public cloud vs on premise?

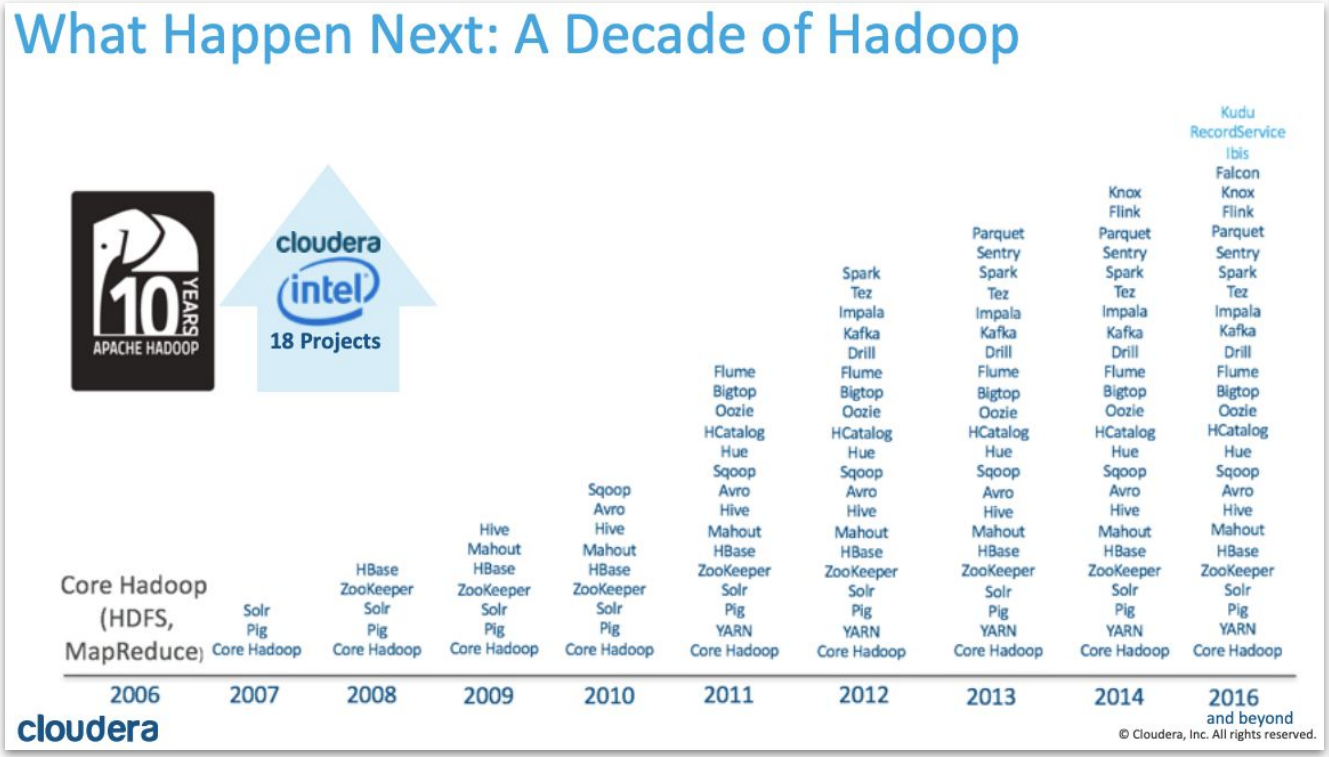
Public cloud

- Can start now
- Elastic with resources and cost
- Inexpensive for small scale
- Expensive for huge scale
- Ease of operations

On premise

- Wait until it is built (a few months)
- Inelastic with resources, fixed cost
- Expensive for small scale
- Relatively inexpensive for huge scale
- Cost of operations

Apache Hadoop ecosystem to build and run a big data platform



Projects by category:

- big-data (49):
  - Apache Accumulo
  - Apache Airavata
  - Apache Ambari
  - Apache Apex (in the Attic)
  - Apache Avro
  - Apache Beam
  - Apache Bigtop
  - Apache BookKeeper
  - Apache Calcite
  - Apache Camel
  - Apache CarbonData
  - Apache CouchDB
  - Apache Crunch
  - Apache Daffodil (Incubating)
  - Apache DataFu (Incubating)
  - Apache DirectMemory (in the Attic)
  - Apache Drill
  - Apache Edgent (Incubating)
  - Apache Falcon (in the Attic)
  - Apache Flink
  - Apache Flume
  - Apache Fluo
  - Apache Fluo Recipes
  - Apache Fluo YARN
  - Apache Giraph
  - Apache Hama
  - Apache Helix
  - Apache Ignite
  - Apache Kibble
  - Apache Knox
  - Apache Kudu
  - Apache Lens
  - Apache MetaModel
  - Apache OODT
  - Apache Oozie
  - Apache ORC
  - Apache Parquet
  - Apache Phoenix
  - Apache PredictionIO
  - Apache REEF
  - Apache Samza
  - Apache Spark
  - Apache Sqoop
  - Apache Storm
  - Apache Tajo
  - Apache Tez
  - Apache Trafodion
  - Apache VXQuery
  - Apache Zeppelin

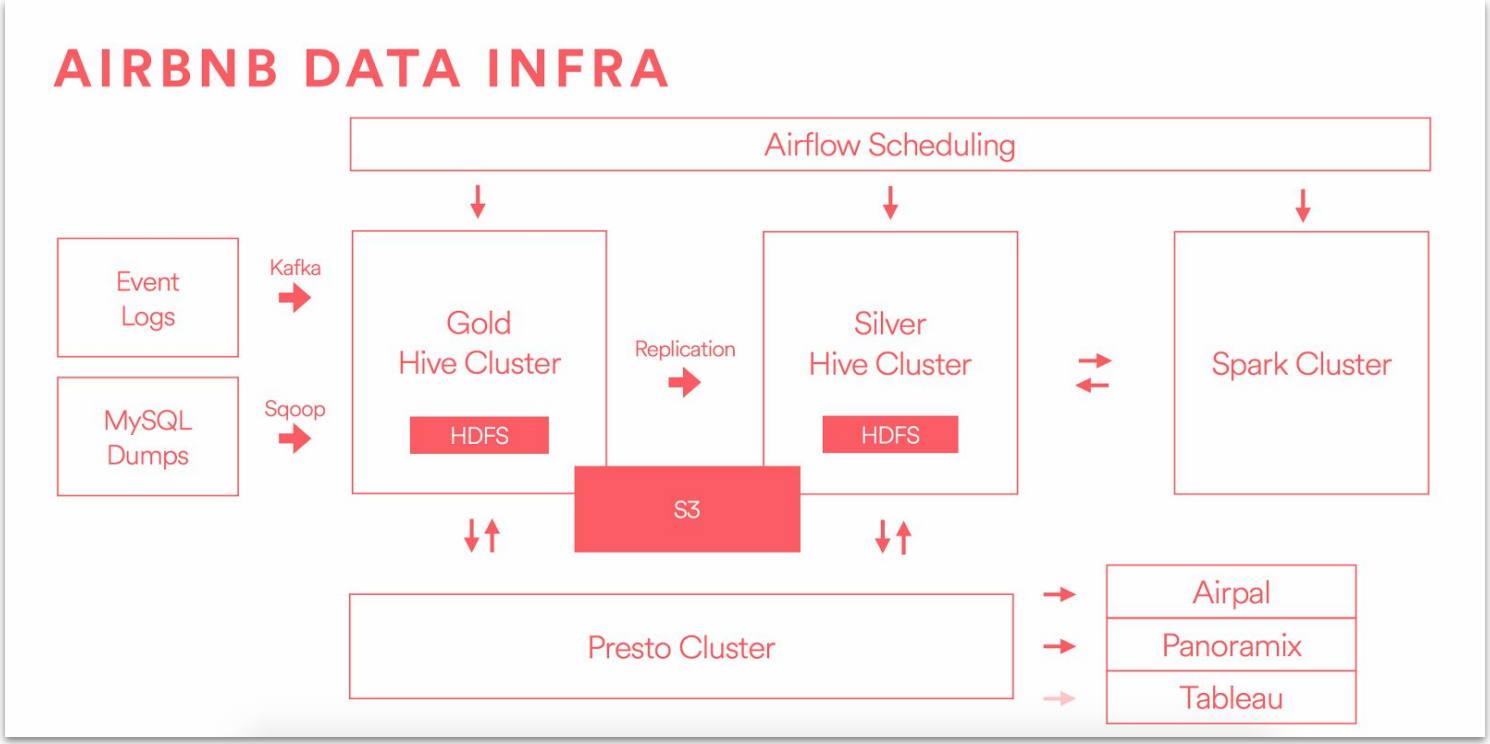
- database (24):
  - Apache Cassandra
  - Apache Cayenne
  - Apache Cocoon
  - Apache CouchDB
  - Apache Curator
  - Apache Derby
  - Apache Empire-db
  - Apache Forrest
  - Apache Gora
  - Apache HBase
  - Apache Hive
  - Apache Ignite
  - Apache Jackrabbit
  - Apache Lucene Core
  - Apache Lucene.Net
  - Apache Lucy (in the Attic)
  - Apache MetaModel
  - Apache OFBiz
  - Apache OpenJPA
  - Apache ORC
  - Apache Phoenix
  - Apache Pig
  - Apache Torque
  - Apache ZooKeeper

https://www.apache.org/

Big data platform architecture: Self built

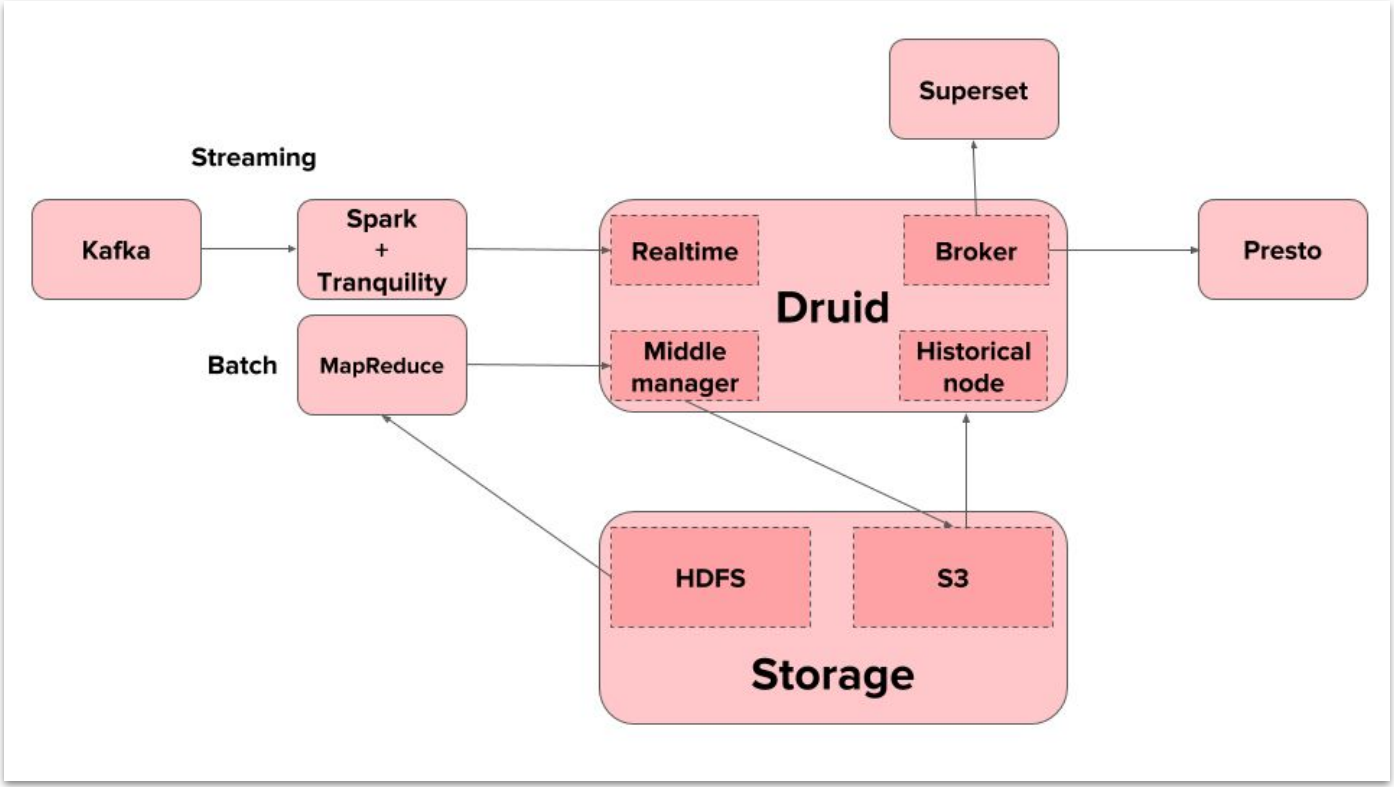
Section 2 of 10

Airbnb: Big data platform (2016)



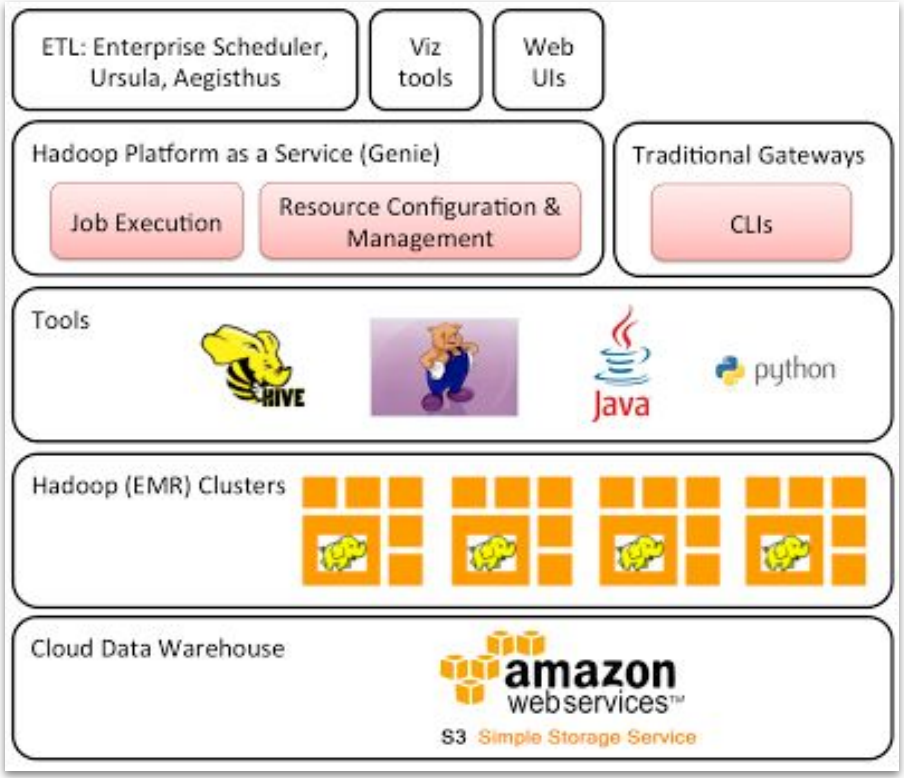
https://medium.com/airbnb-engineering/data-infrastructure-at-airbnb-8adfb34f169c

Airbnb: Big data platform (2018)



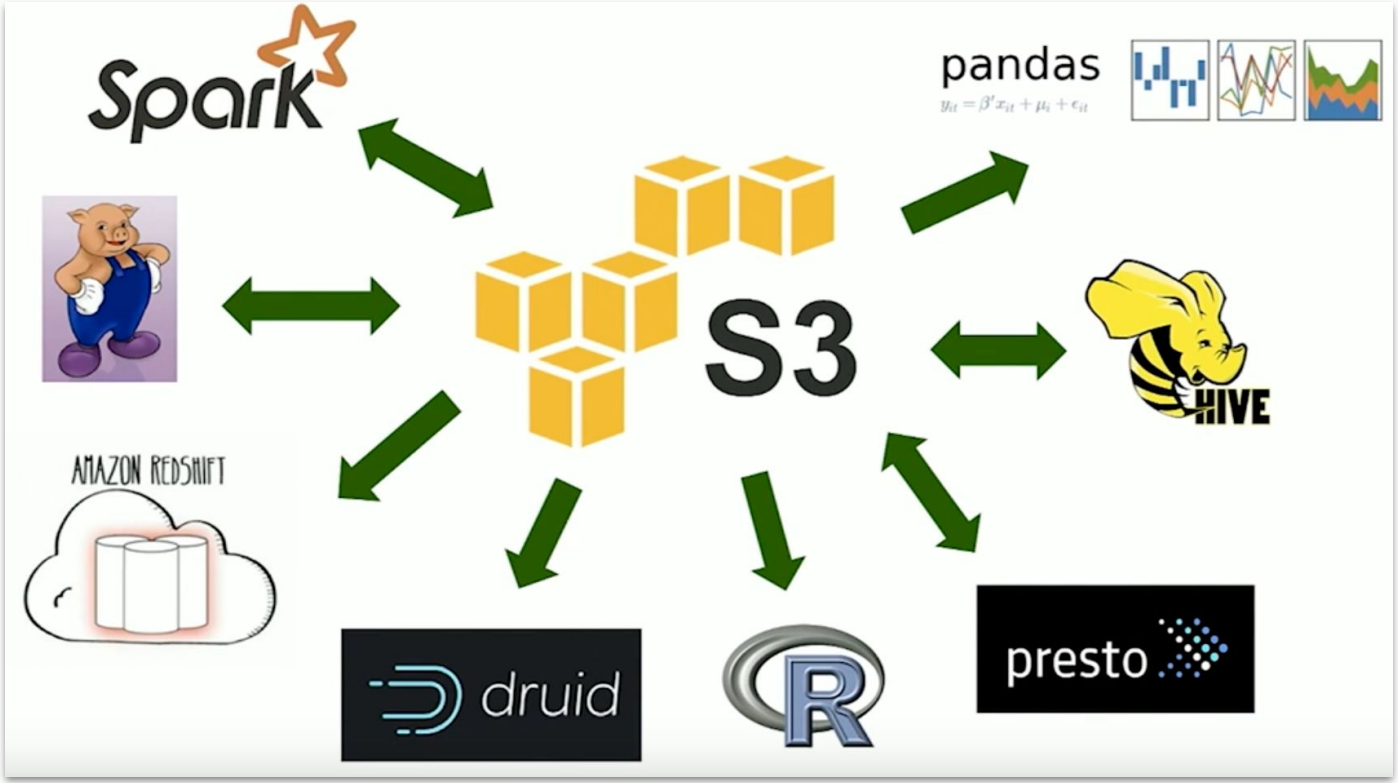
https://medium.com/airbnb-engineering/druid-airbnb-data-platform-601c312f2a4c

Netflix: Big data platform (2013)



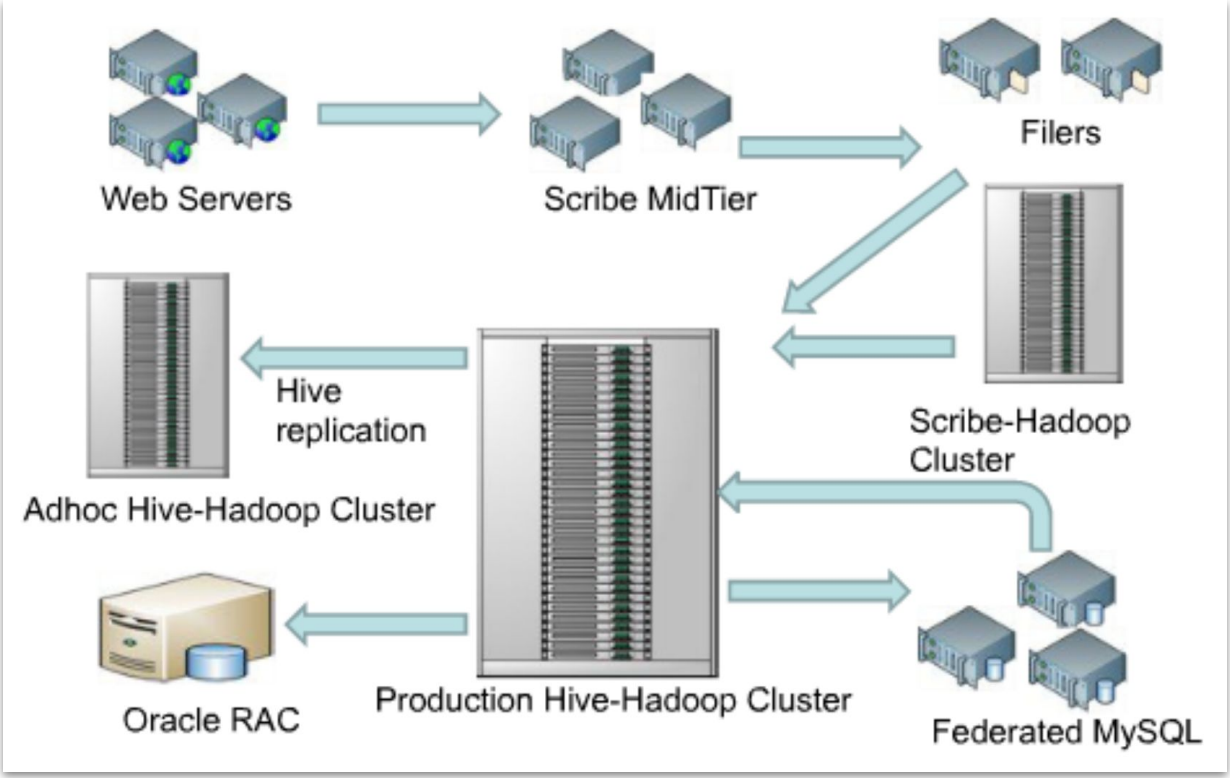
https://medium.com/netflix-techblog/hadoop-platform-as-a-service-in-the-cloud-c23f35f965e7

Netflix: Big data platform (2017)



https://www.youtube.com/watch?v=CSDIThSwA7s

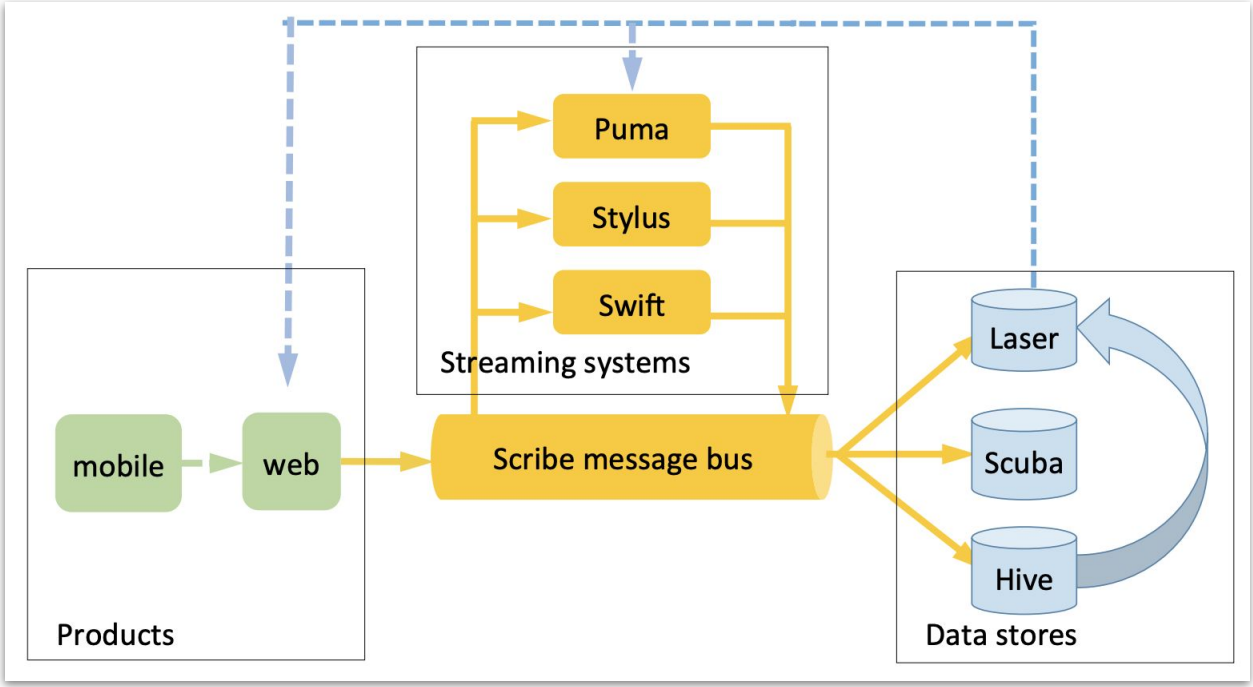
Facebook: Big data platform (2008)



https://research.fb.com/wp-content/uploads/2010/06/data-warehousing-and-analytics-infrastructure-at-facebook.pdf

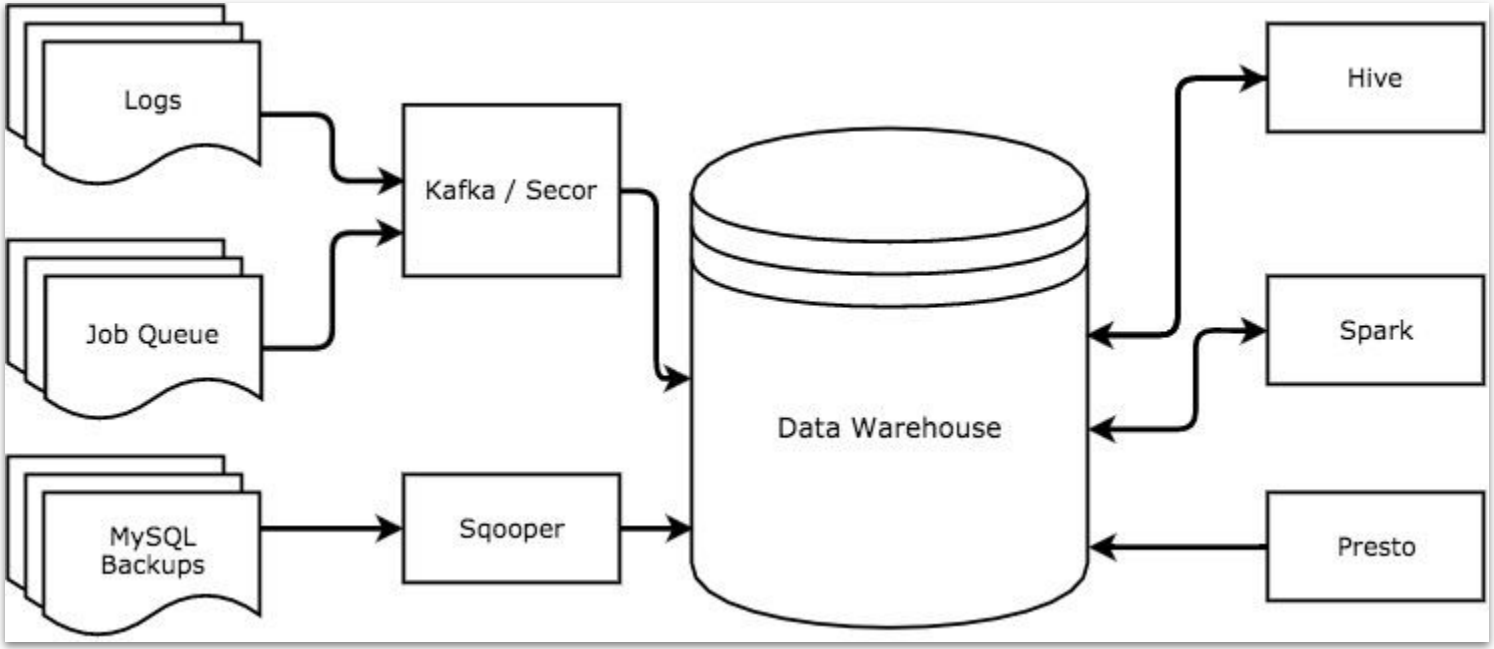
Facebook: Real-time processing (2016)

- Scribe: Message bus, transport mechanism for sending data to batch and real-time systems
- Puma: Stream processing system with apps written in a SQL-like language
- Swift: Stream processing engine that provides checkpointing for Scribe
- Stylus: Low-level stream processing framework
- Laser: K/V store built on RocksDB
- Scuba: Slice-and-dice analysis data store
- Hive: Exabyte-scale data warehouse



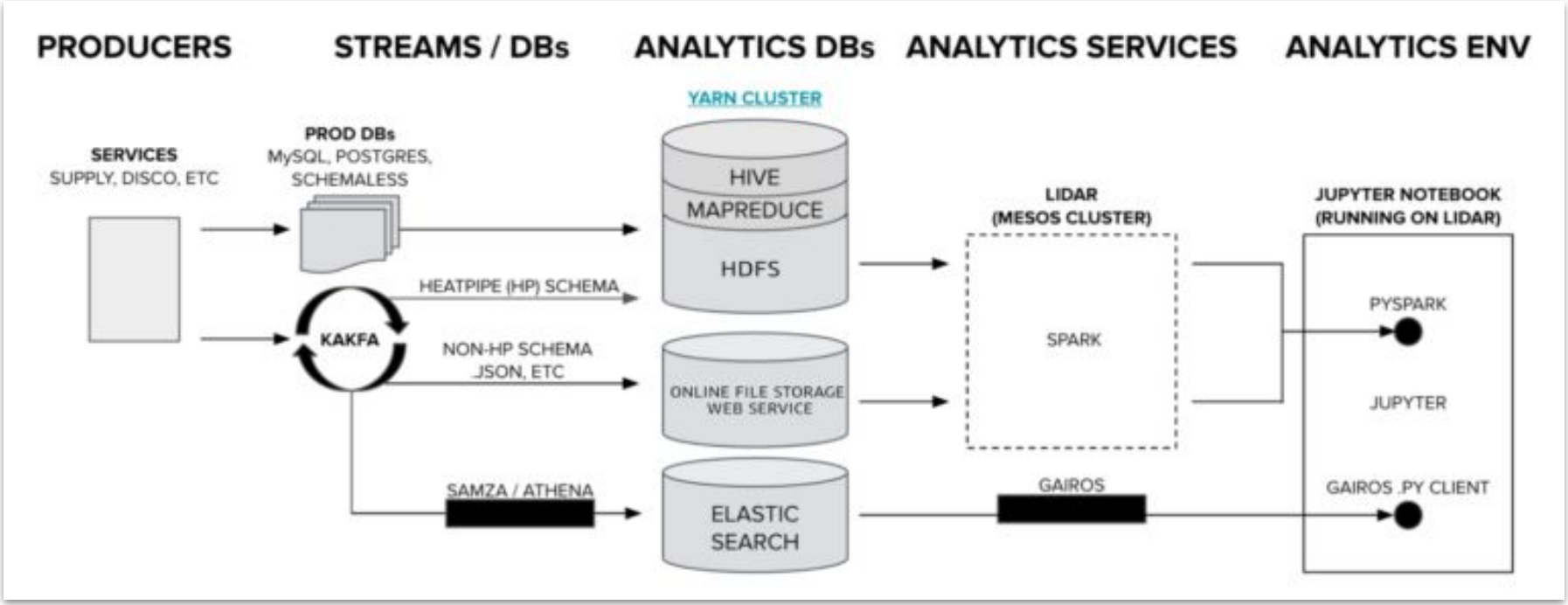
https://research.fb.com/wp-content/uploads/2016/11/realtime_data_processing_at_facebook.pdf

Slack: Big data platform (2016)



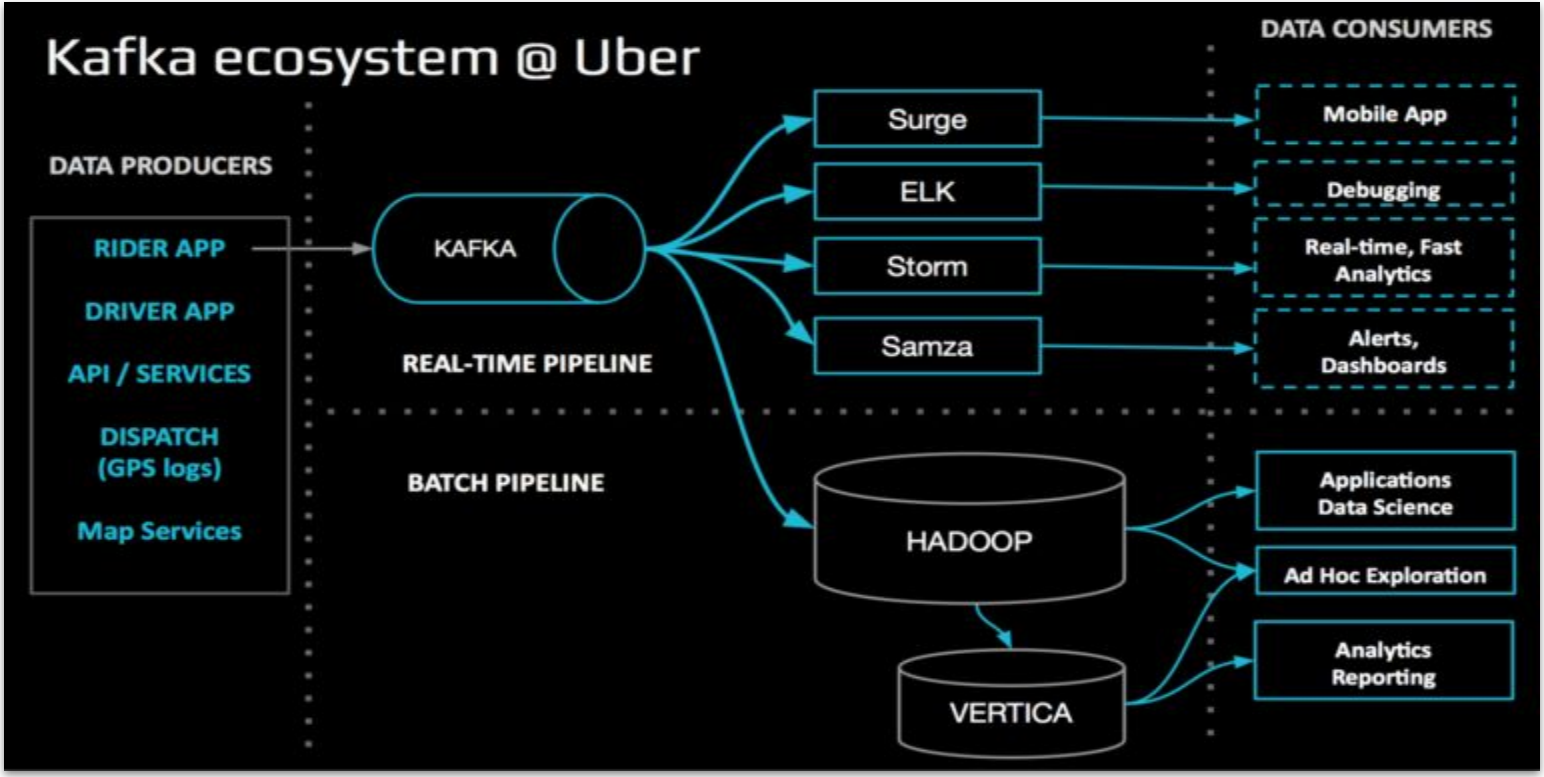
https://slack.engineering/data-wrangling-at-slack-f2e0ff633b69

Uber: Big data platform (2016)



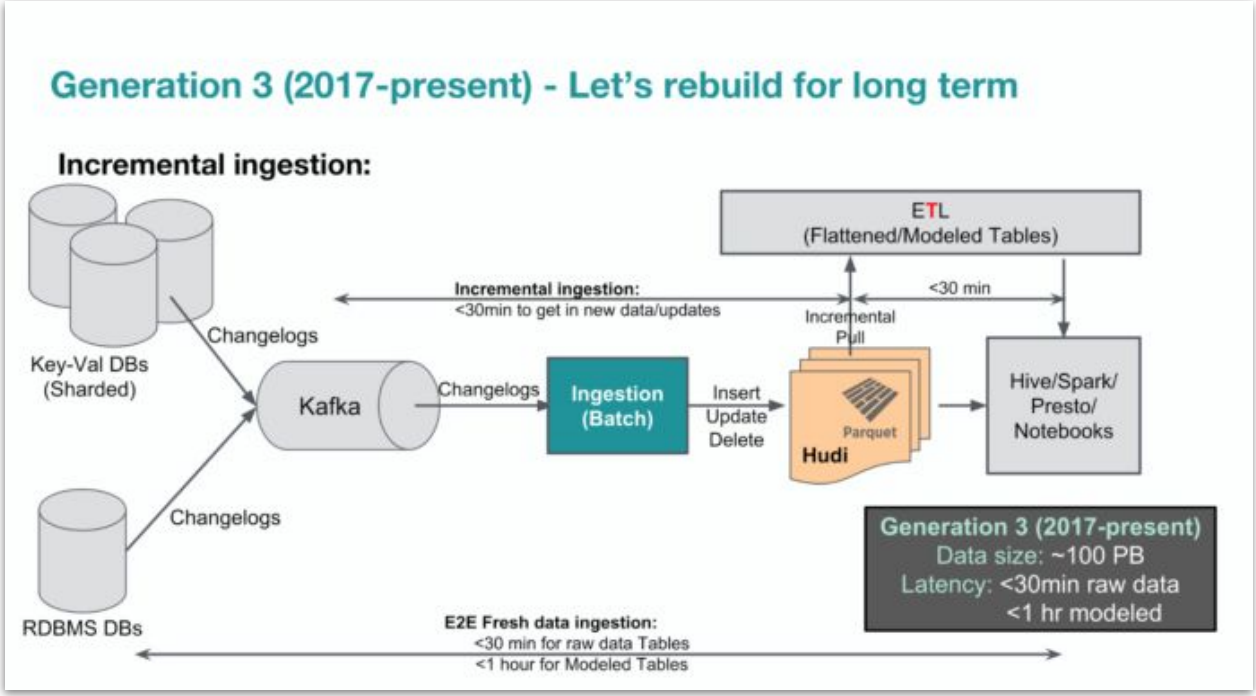
https://eng.uber.com/tech-stack-part-two/

Uber: Real-time processing (2016)



https://eng.uber.com/ureplicator/

Uber: Big data platform (2019)



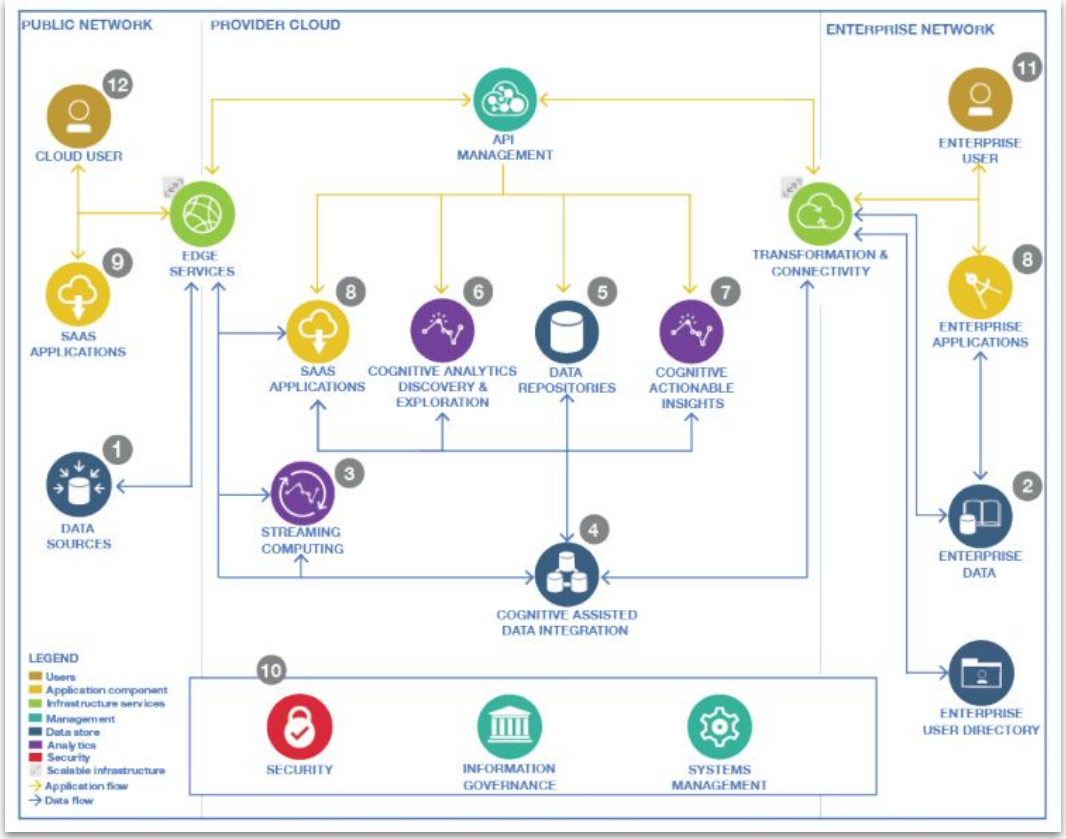
"The third generation of our Big Data platform incorporates faster, incremental data ingestion (using our open source Marmaray framework), as well as more efficient storage and serving of data via our open source Hudi library."

https://eng.uber.com/uber-big-data-platform/

Big data platform architecture: Cloud built

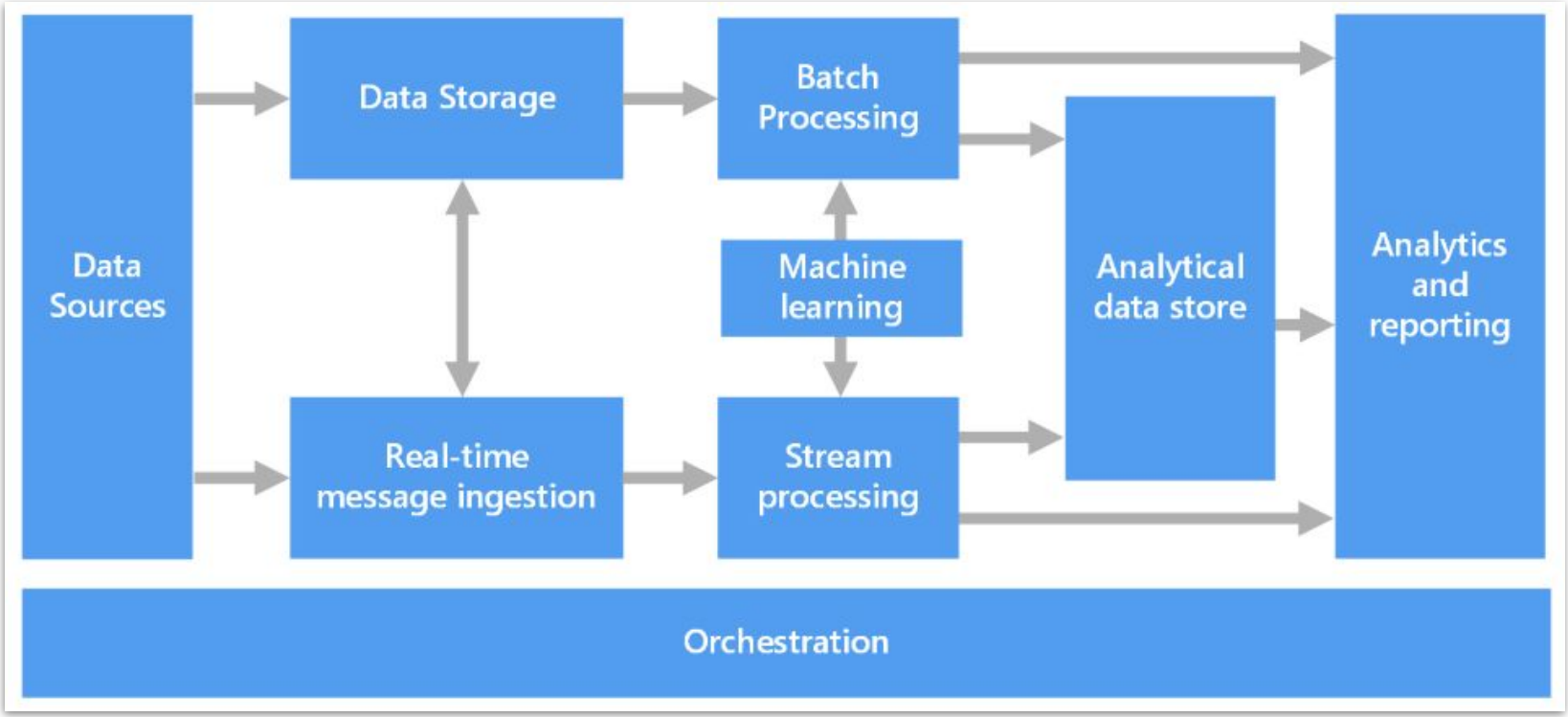
Section 2 of 10

IBM Cloud: Big data platform



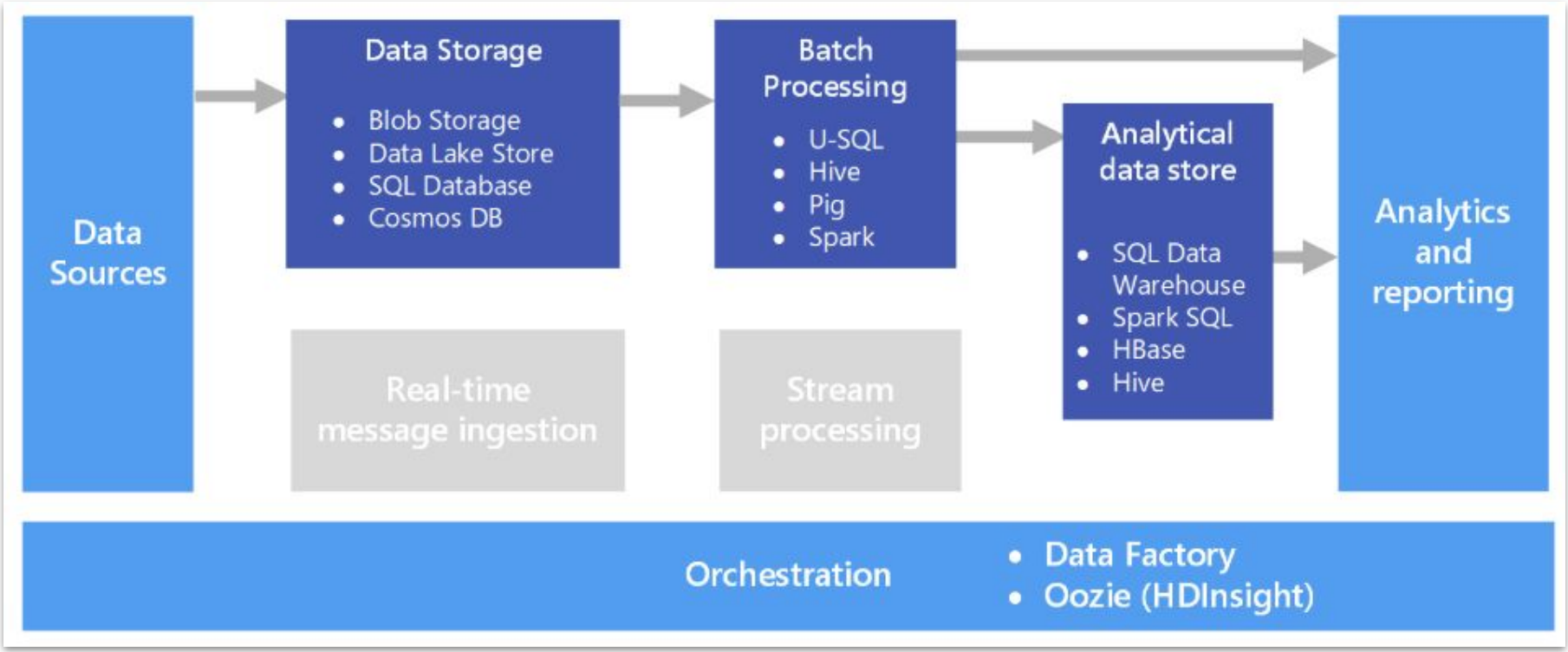
https://www.ibm.com/cloud/garage/architectures/dataAnalyticsArchitecture/reference-architecture

Microsoft Cloud: Big data platform



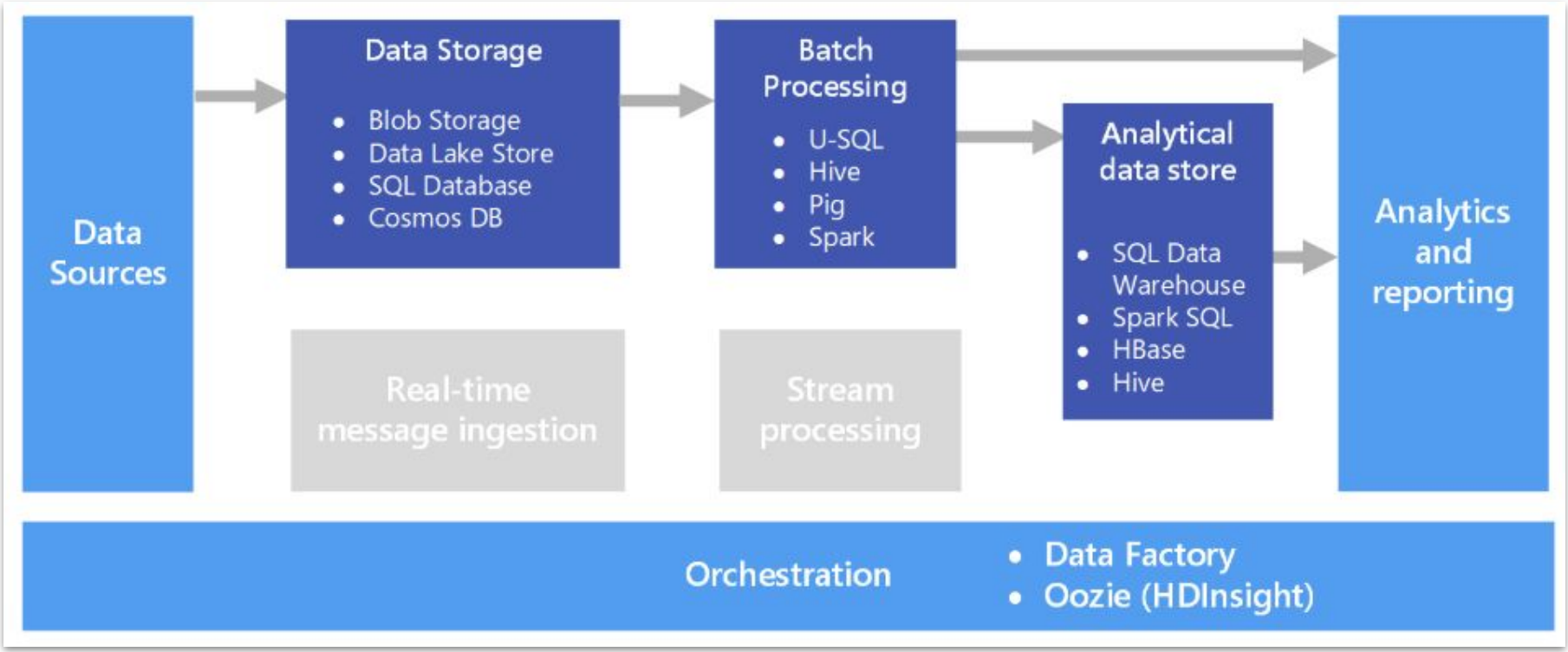
https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

Microsoft Cloud: Batch processing



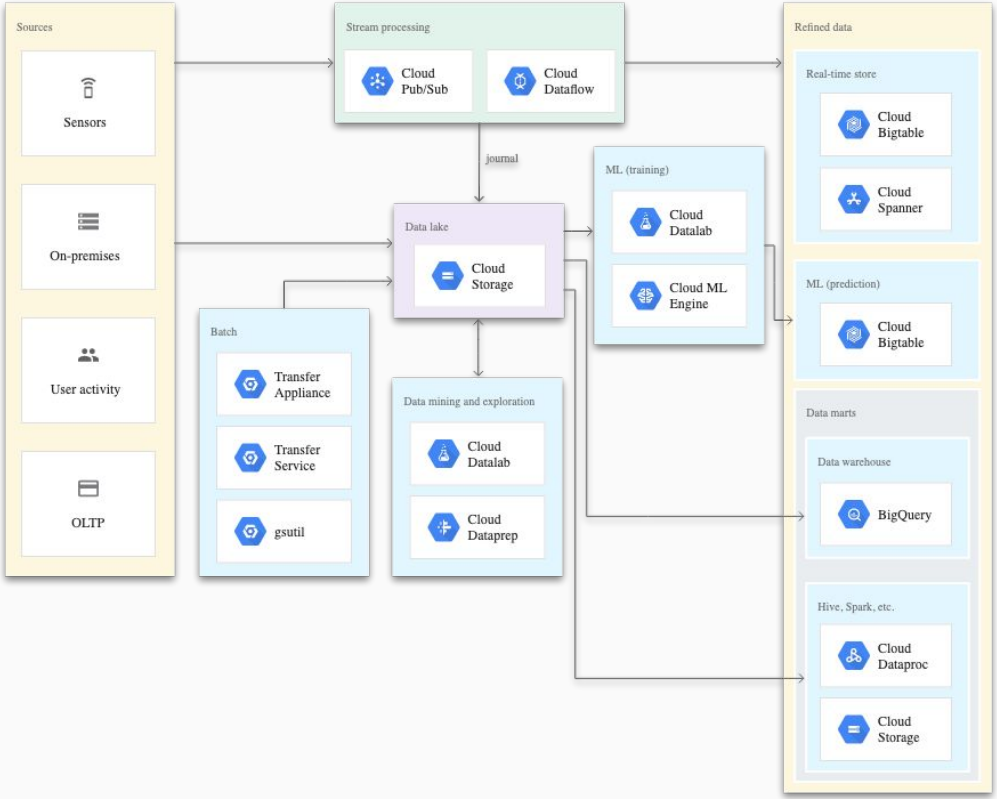
https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

Microsoft Cloud: Real-time streaming



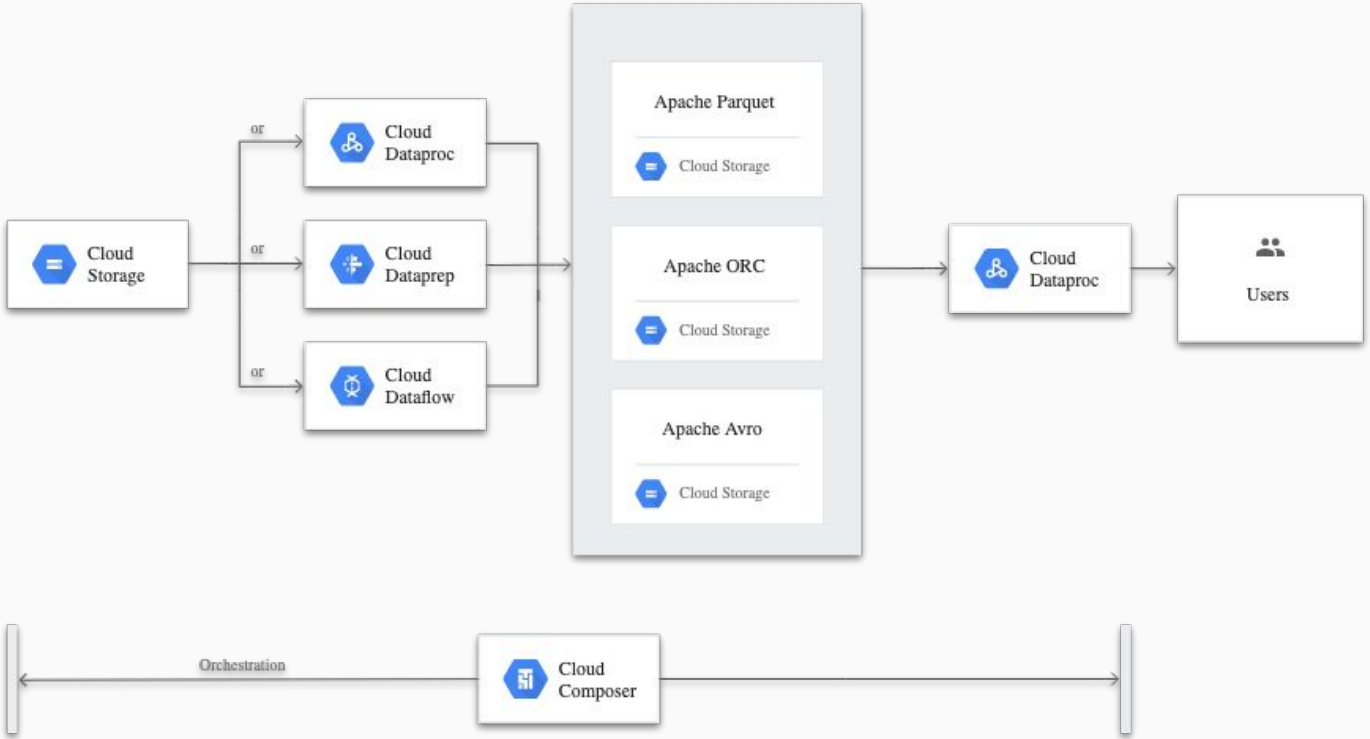
https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

Google Cloud: Big data platform



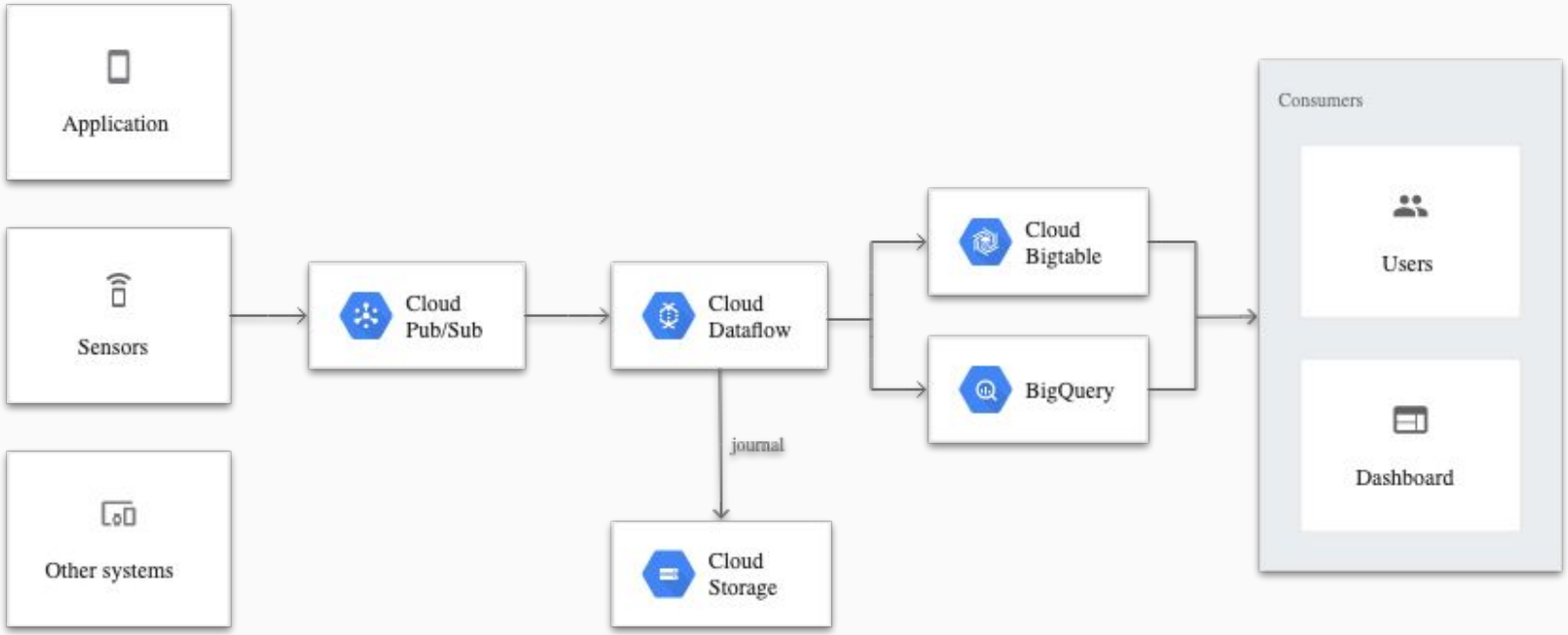
https://cloud.google.com/solutions/build-a-data-lake-on-gcp

Google Cloud: Batch processing



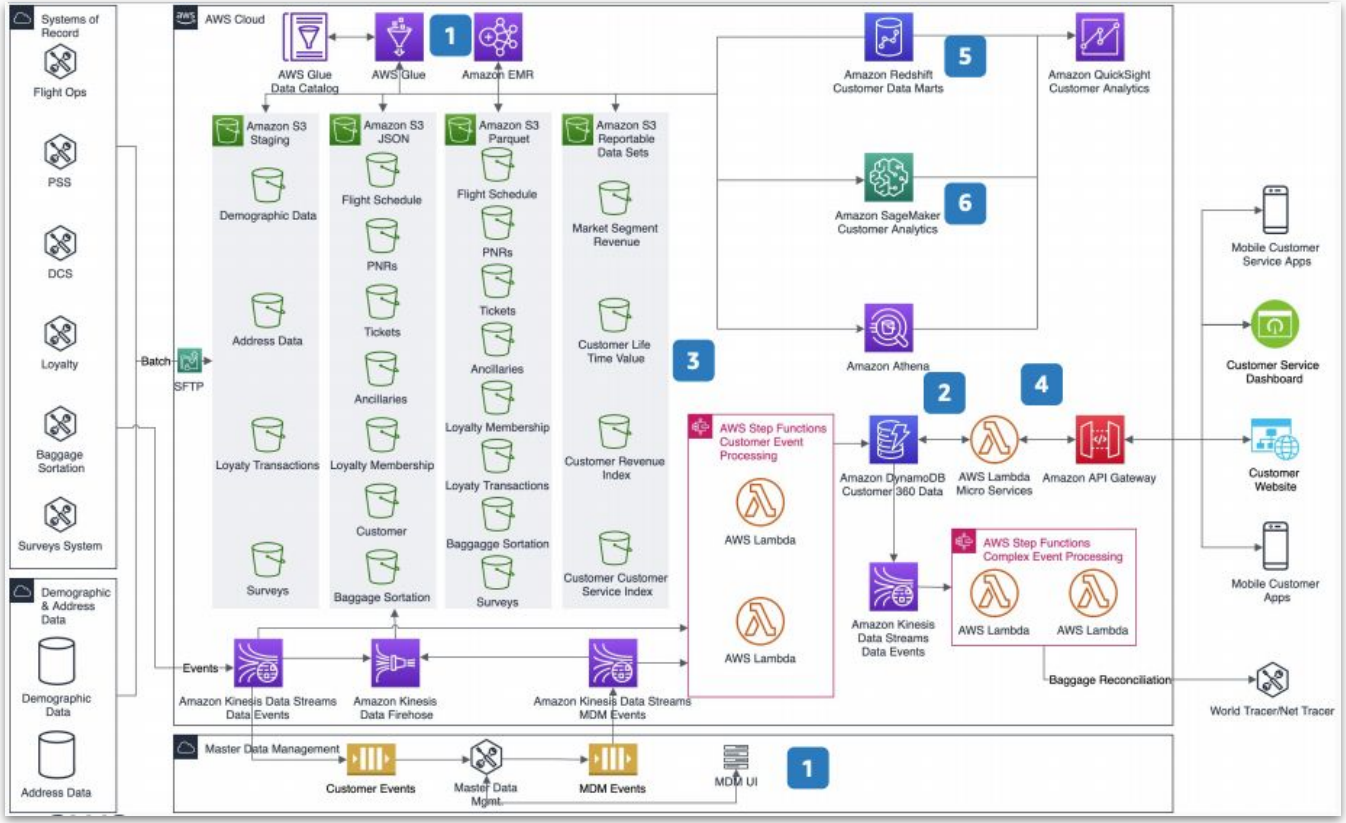
https://cloud.google.com/solutions/build-a-data-lake-on-gcp

Google Cloud: Real-time processing



https://cloud.google.com/solutions/build-a-data-lake-on-gcp

Amazon AWS: Big data platform



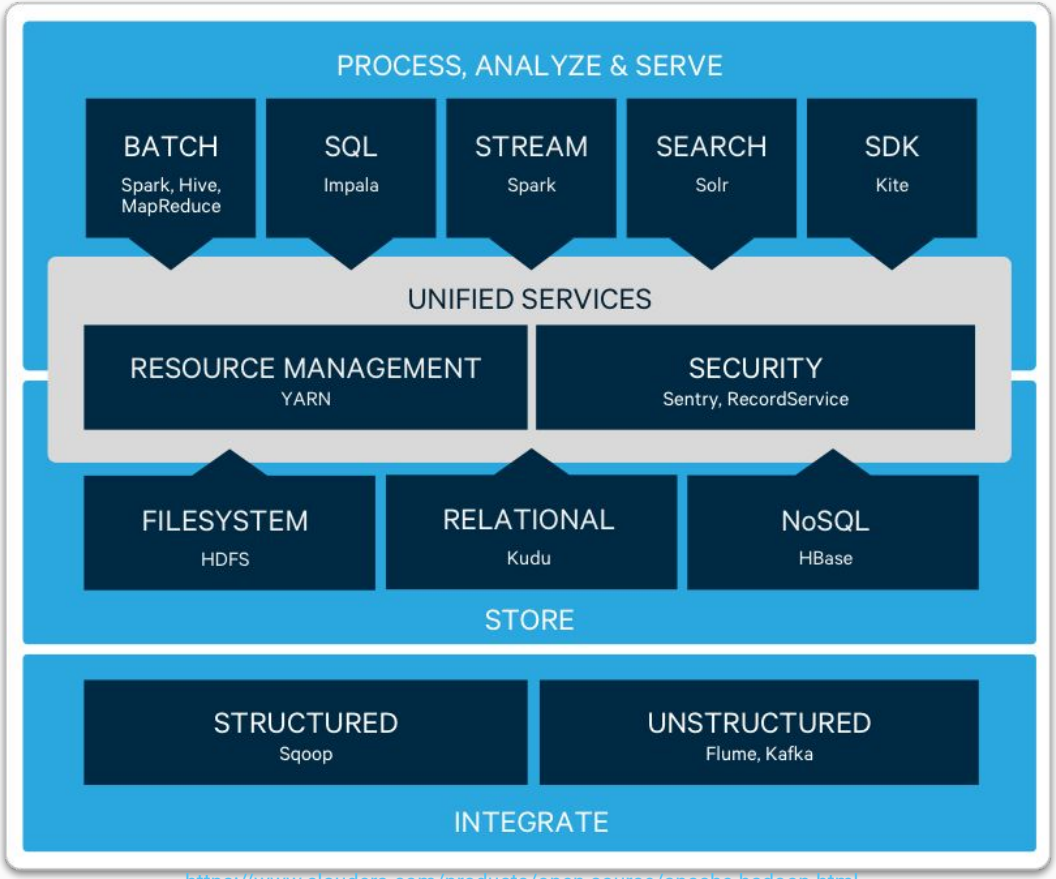
1. Augment the single view of customer data as a service by using an MDM tool to create a customer master. A simple MDM with customer identification and de-duplications can also be built using ML Transform in **AWS Glue**. This machine learning transformation identifies duplicates in loyalty membership and customers that have not signed up for loyalty membership. This information allows for clearly identifying customers that have significant travel beyond loyalty members.
2. The operations data stores, data lakes, and analytics platforms must be adapted to new customer data feeds. New capabilities will be delivered as new versions of customer-centric microservices and events are developed.
3. The data lake is enhanced by changing to a customer centric view for lifetime value, revenue index, and service index.
4. Agile development teams can quickly add new microservices to consume and publish the new customer centric services.
5. Agile data teams can quickly augment the EDW with new customer centric domain schemas and data marts.
6. Data scientists can use existing raw and curated data as well as new customer data to build new models in **Amazon SageMaker**.

https://d1.awsstatic.com/architecture-diagrams/ArchitectureDiagrams/svoc_data_platform_mdm_ra.pdf?did=wp_card&trk=wp_card

Big data platform architecture: Consulting built

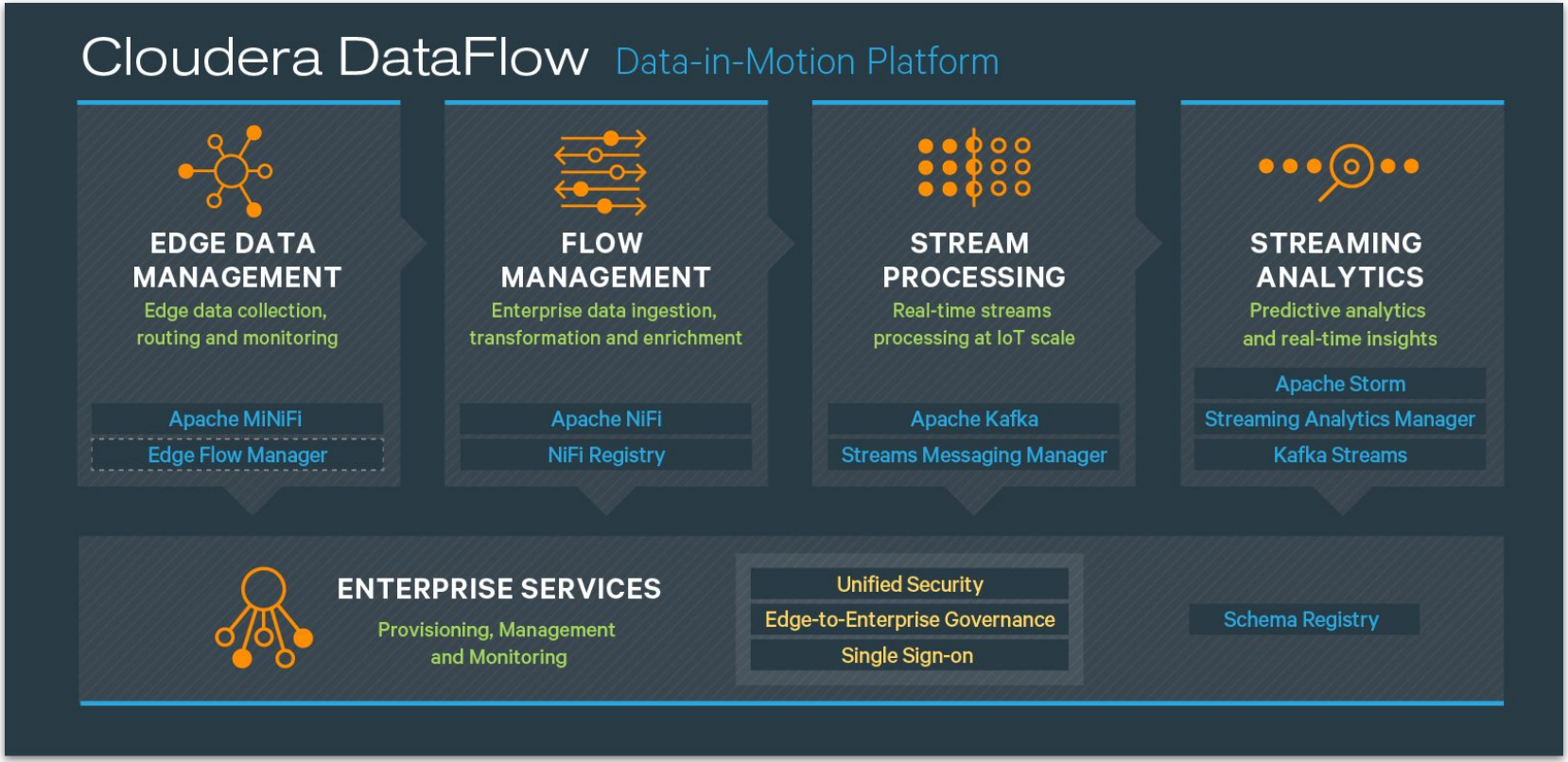
Section 2 of 10

Cloudera: Big data using Hadoop ecosystem



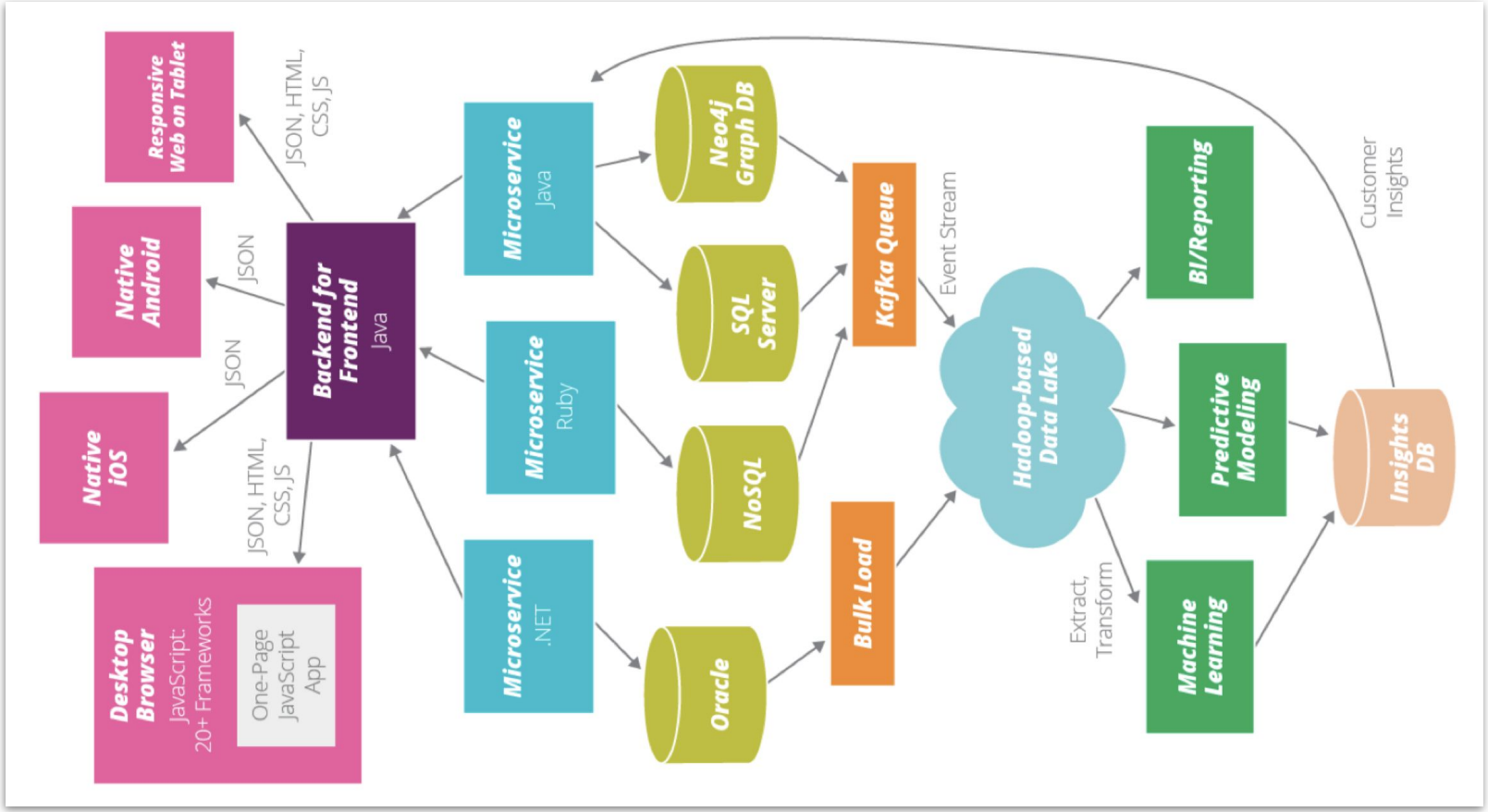
https://www.cloudera.com/products/open-source/apache-hadoop.html

Cloudera: Real-time processing

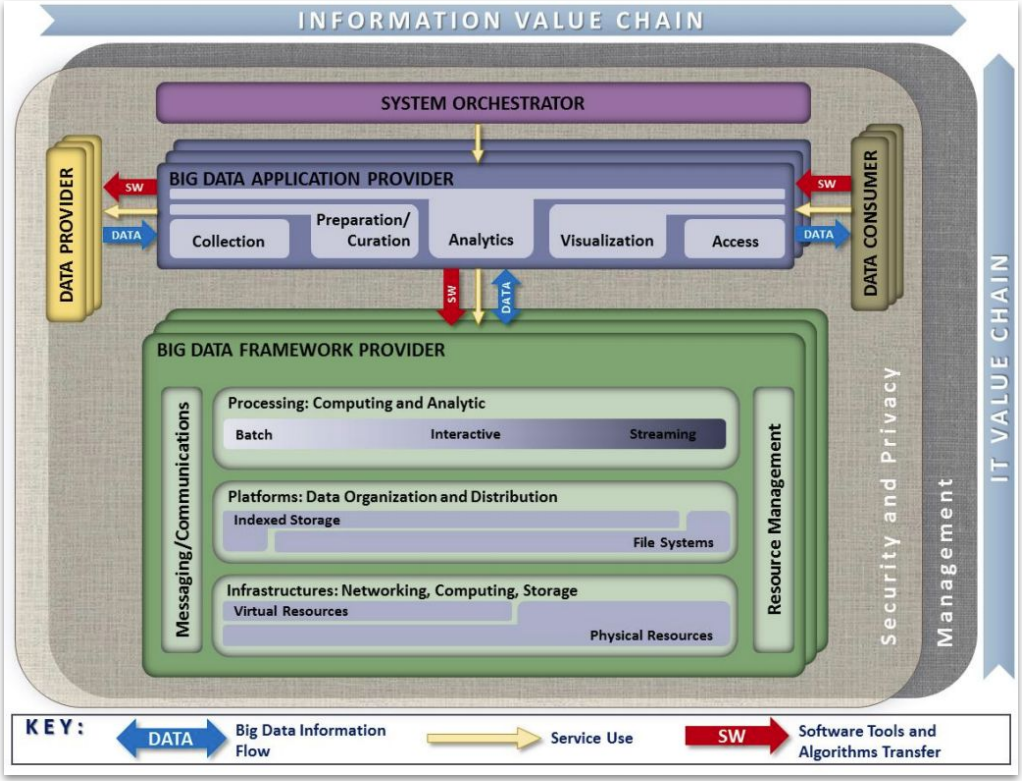


https://blog.cloudera.com/introducing-cloudera-dataflow-cdf/

ThoughtWorks: Big data platform

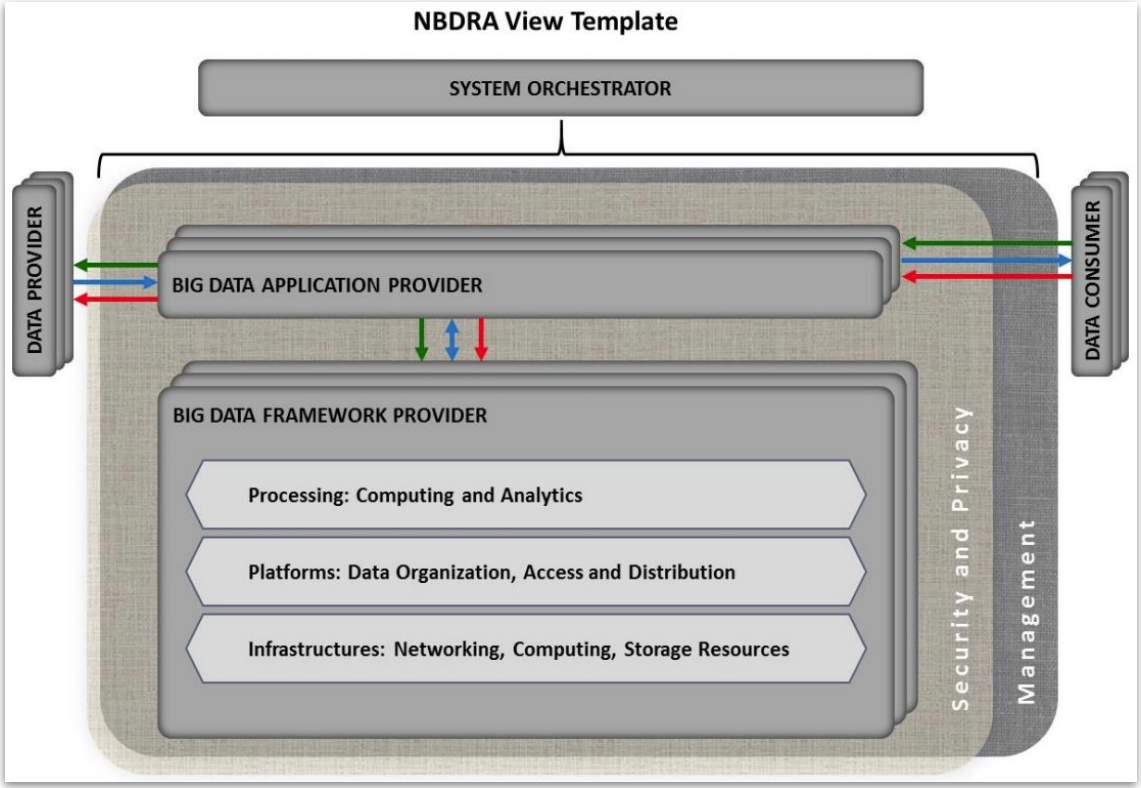


NIST Big data reference architecture (NBDRA)



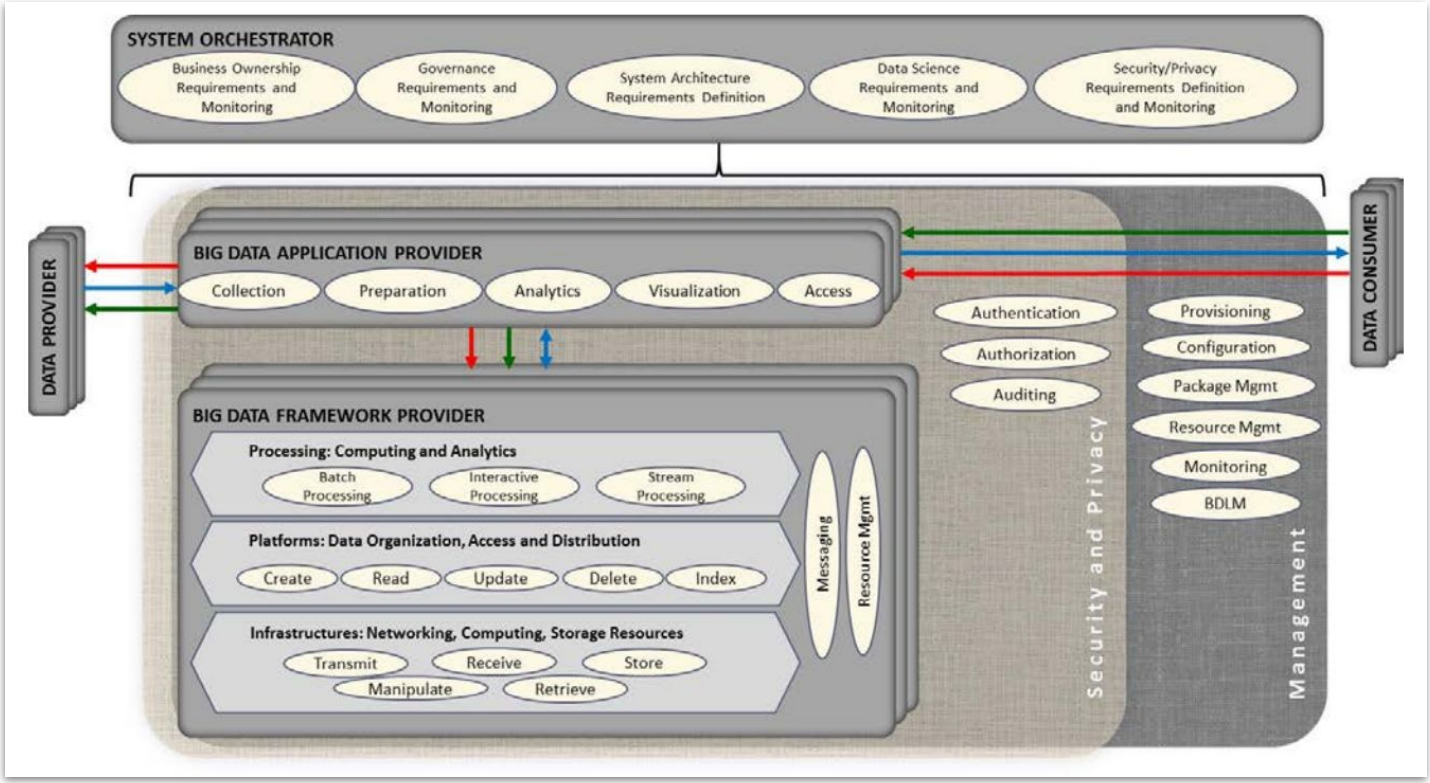
https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-6r1.pdf

NIST NBDRA: Top level roles and fabrics



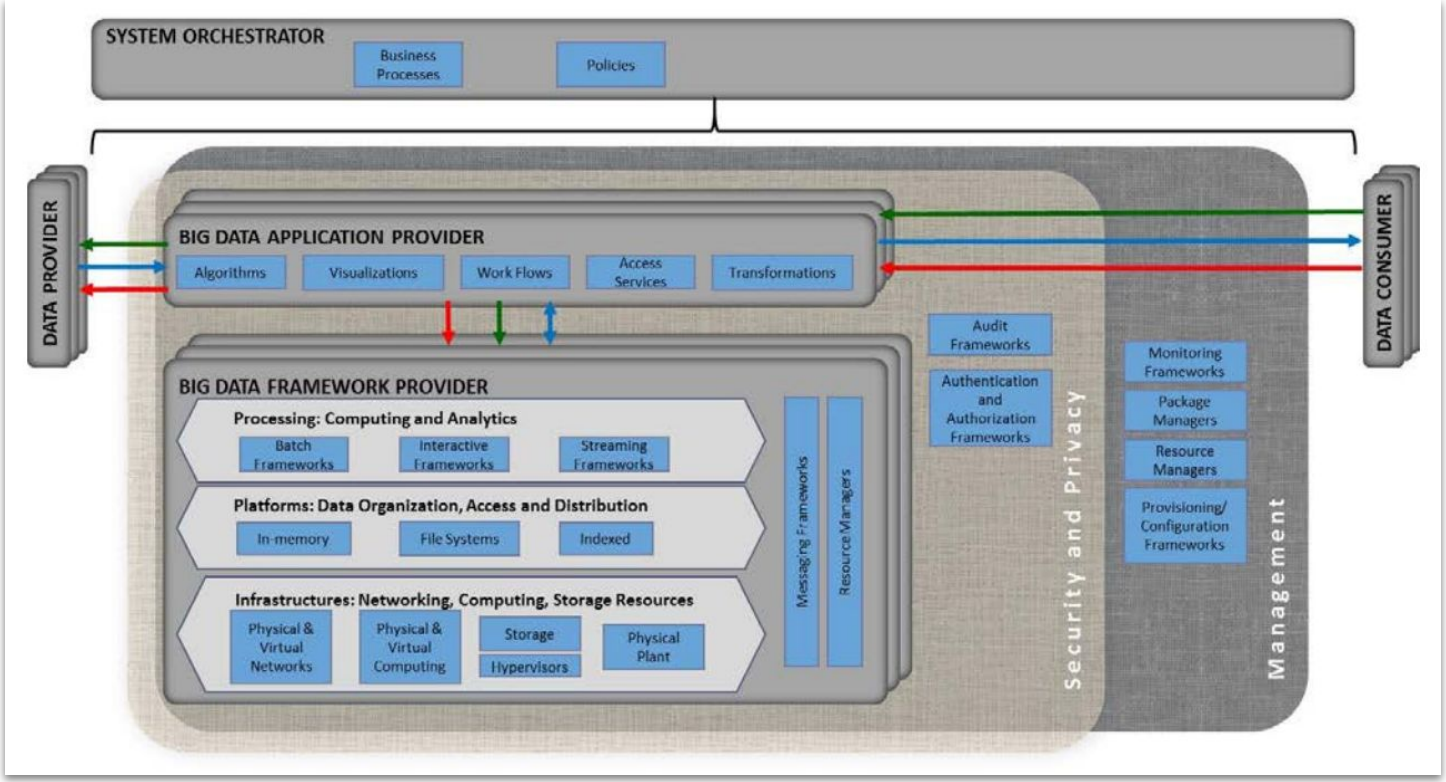
https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-6r1.pdf

NIST NBDRA: Activities view



https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-6r1.pdf

NIST NBDRA: Functional view

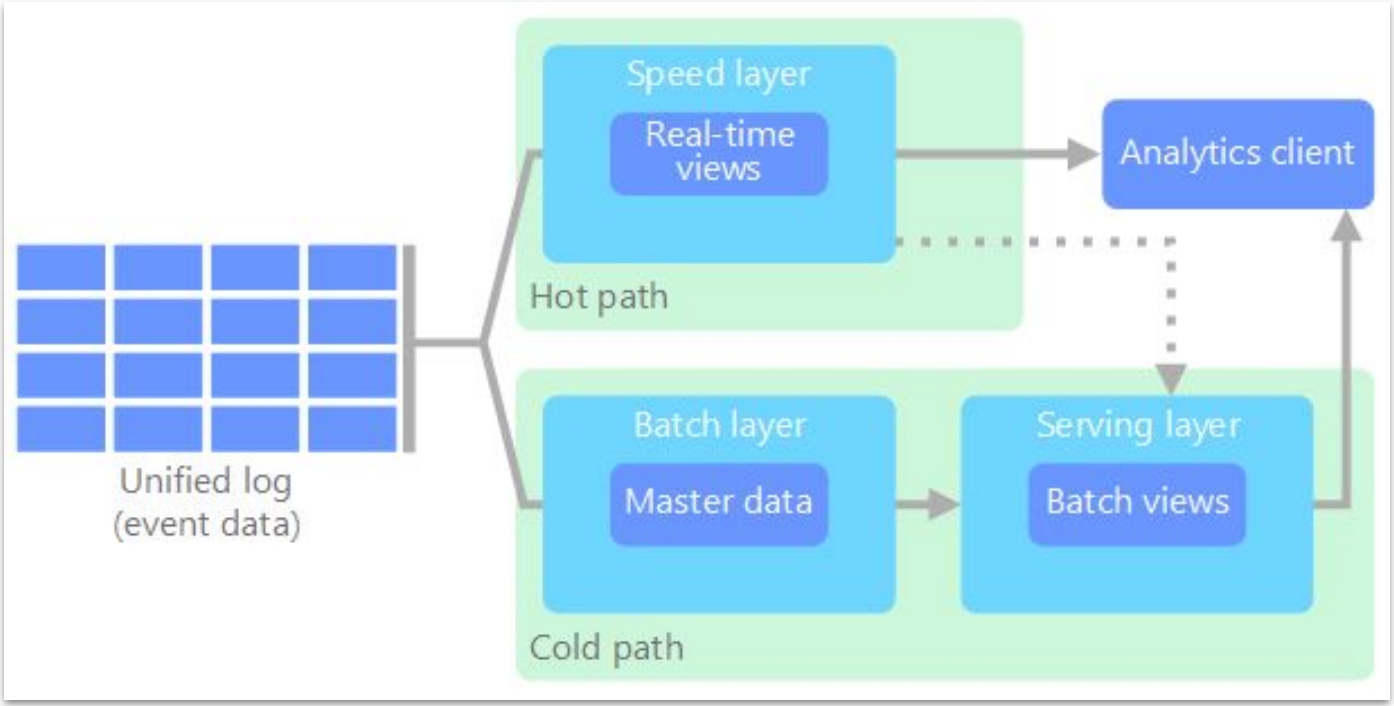


https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-6r1.pdf

Big data platform architecture: Batch vs Streaming

Section 2 of 10

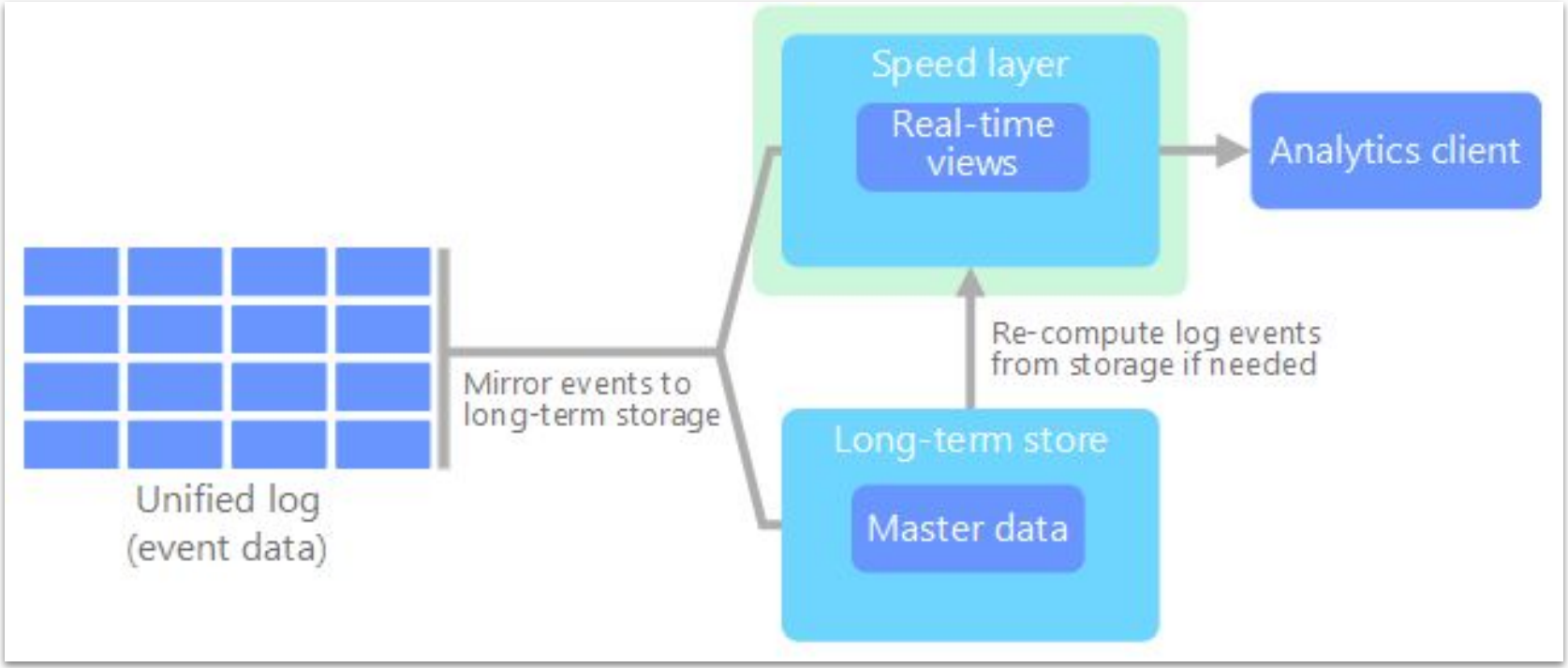
Generic: Lambda architecture



Nathan Marz, creator of Storm

https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

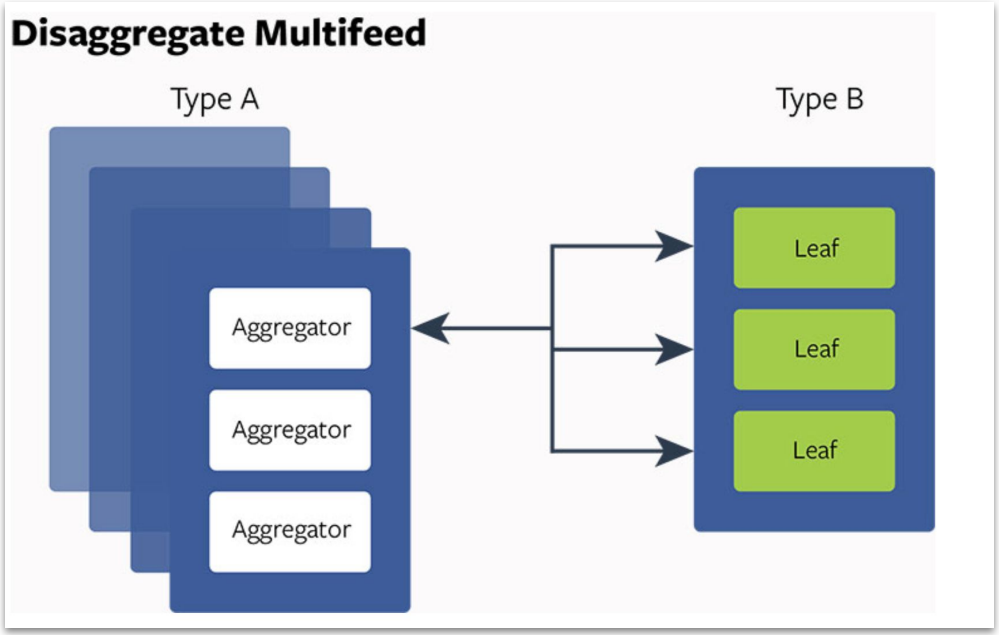
Generic: Kappa architecture

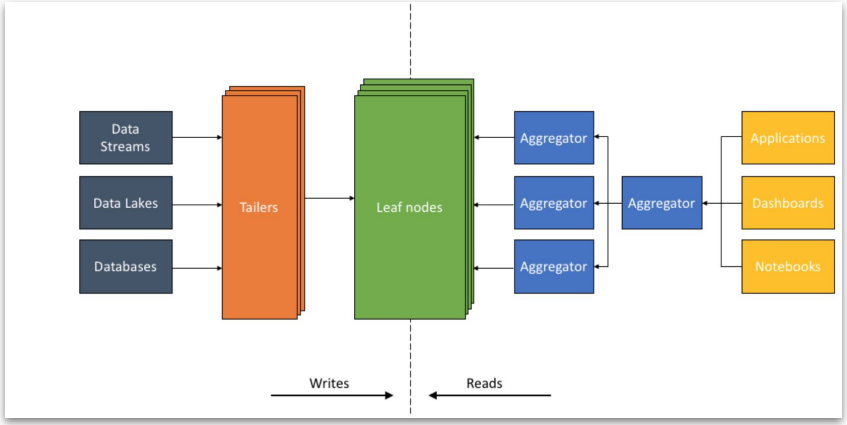


Jay Kreps, creator of Kafka

https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

Facebook: Aggregator Leaf Tailer Architecture





https://engineering.fb.com/production-engineering/serving-facebook-multifeed-efficiency-performance-gains-through-redesign/

Streaming: Design decisions and their impact

- Language paradigms
  - Declarative, functional, procedural
- Data transfer between nodes of the DAG
  - Direct
  - Broker based
  - Persistent storage based
- Processing (state vs output) semantics
  - At least once, at most once, exactly once
- State-saving mechanism
  - Replication, persistence via local DB or remote DB, upstream backup, global consistent snapshot
- Reprocessing
  - Stream only, separate stream and batch, stream also as batch

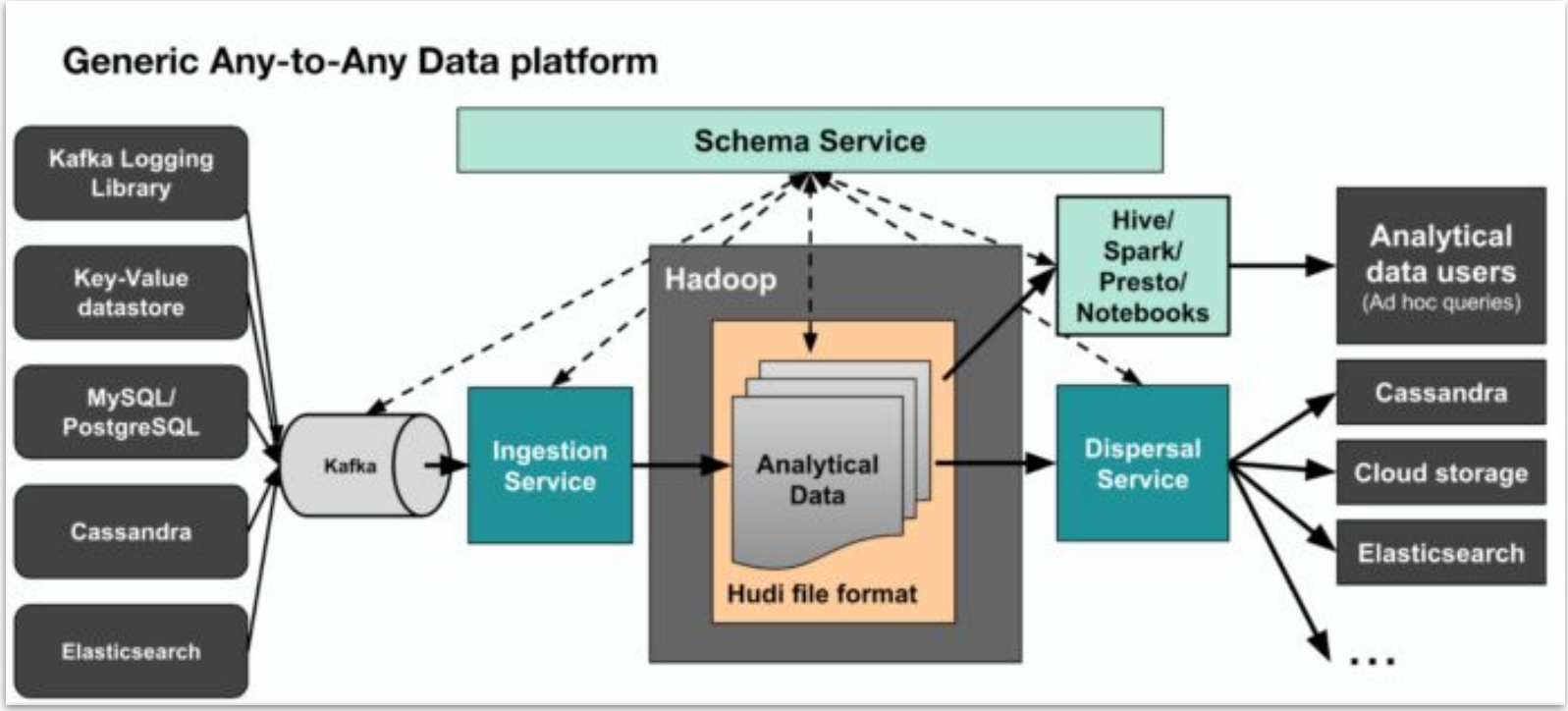
Design decision	Ease of use	Performance	Fault tolerance	Scalability	Correctness
Language paradigm	X	X			
Data transfer	X	X	X	X	
Processing semantics			X		X
State-saving mechanism	X	X	X	X	X
Reprocessing	X			X	X

Design decision	Puma	Stylus	Swift	Storm	Heron	Spark Streaming	Millwheel	Flink	Samza
Language paradigm	SQL	C++	Python	Java	Java	Functional	C++	Functional	Java
Data transfer	Scribe	Scribe	Scribe	RPC	Stream Manager	RPC	RPC	RPC	Kafka
Processing semantics	at least	at least at most exactly	at least at most	at least	at least	best effort exactly	at least exactly	at least exactly	at least
State-saving mechanism	remote DB	local DB remote DB				limited	remote DB	global snapshot	local DB
Reprocessing	same code	same code	no batch	same DSL	same DSL	same code	same code	same code	no batch

	State semantics		
Output semantics	At-least-once	At-most-once	Exactly-once
At-least-once	X		X
At-most-once		X	X
Exactly-once			X

https://research.fb.com/wp-content/uploads/2016/11/realtime_data_processing_at_facebook.pdf

Uber: Generic big data architecture for future



"Building a more extensible data transfer platform allowed us to easily aggregate all data pipelines in a standard way under one service as well as support any-to-any connectivity between any data source and data sink."

https://eng.uber.com/uber-big-data-platform/

Conclusion on batch vs streaming

In an established company interfacing with the systems of many partners, it is important to support both batch and streaming for data transfer and operations.

The main reason is that this company needs to adapt to the available external interfaces and get the job done, as it will be almost impossible to force every external interface to adapt to what this company wants to support.

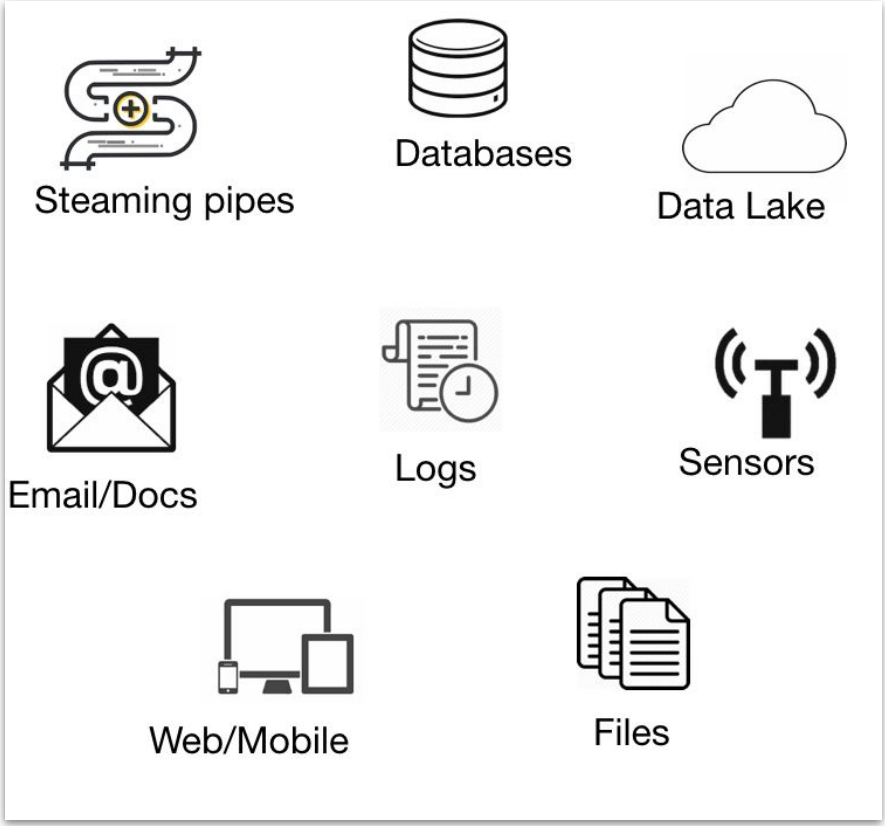
One way to minimize effort is to ensure an application can run both in batch and streaming mode to a large extent.

Having said that, the future is far more real-time, i.e., far more streaming but it may take years for some companies.

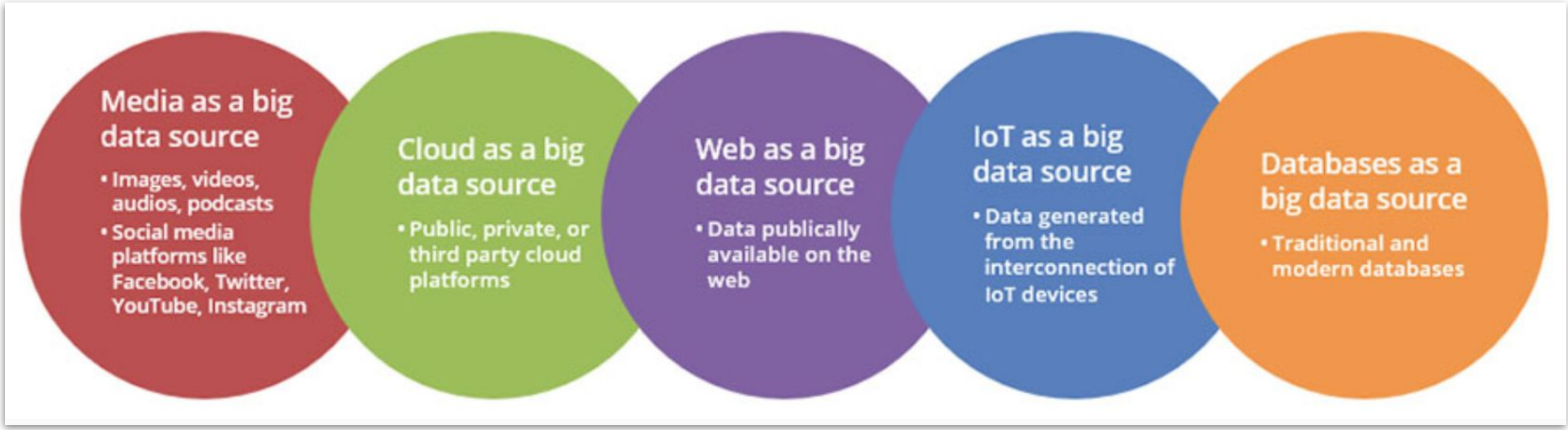
Input layer

Section 3 of 10

Input sources



Top 5 sources of input

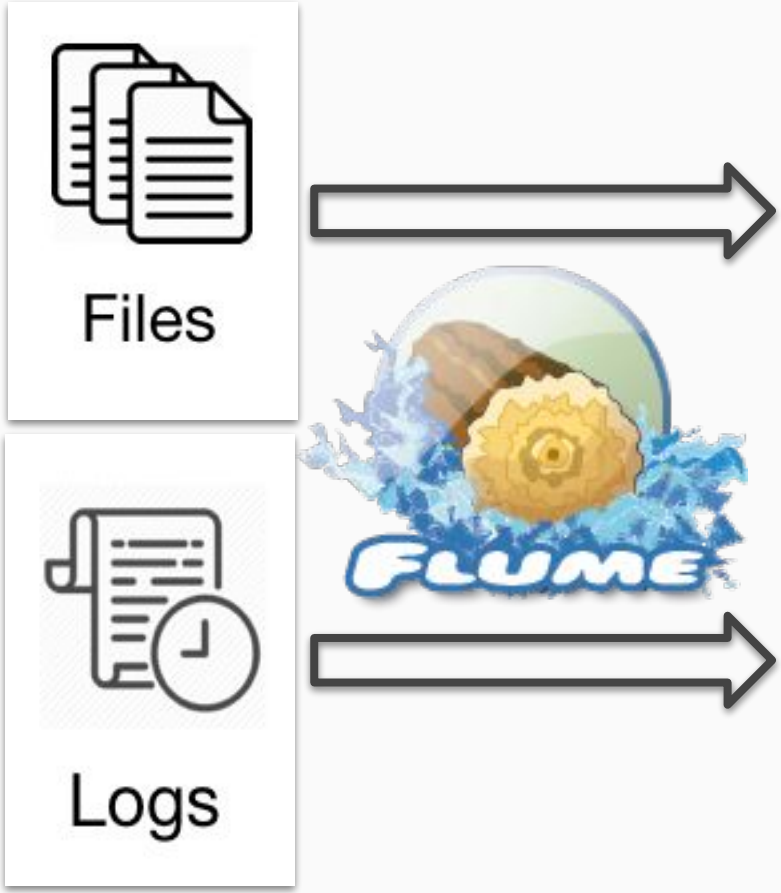


https://www.allerin.com/blog/top-5-sources-of-big-data

Getting data from a database

- Open source tools
  - Apache Sqoop
  - Debezium
- Commercial tools
  - Elastic Beats
  - ??
- Public cloud tools
  - Internal

Getting files / logs



- Stream the file into a streaming system
  - Apache Kafka
- Get files / logs directly
  - Apache Flume
- Public cloud tools
  - Internal

Getting from streaming

- Open source tools
  - Apache Flume
  - Kafka connector for HDFS
  - Kafka connector to <output>
- Public cloud tools
  - Internal

Getting data in batch vs streaming

Batch

- Volume: Huge
- Velocity: Non-real time
- Variety: N/A
- Cost: Lower

Streaming

- Volume: Medium to large
- Velocity: Real time or near real time
- Variety: N/A
- Cost: Higher

Storage layer

Section 4 of 10

Choices for public cloud vs on-prem

Public cloud

- AWS
  - S3
- Google Public Cloud
  - Google Cloud Storage
- Azure
  - Azure Storage
- Also see other vendors

On premise

- Apache Hadoop Distributed File System (HDFS)

Public cloud vs on-prem storage only comparison (2017)

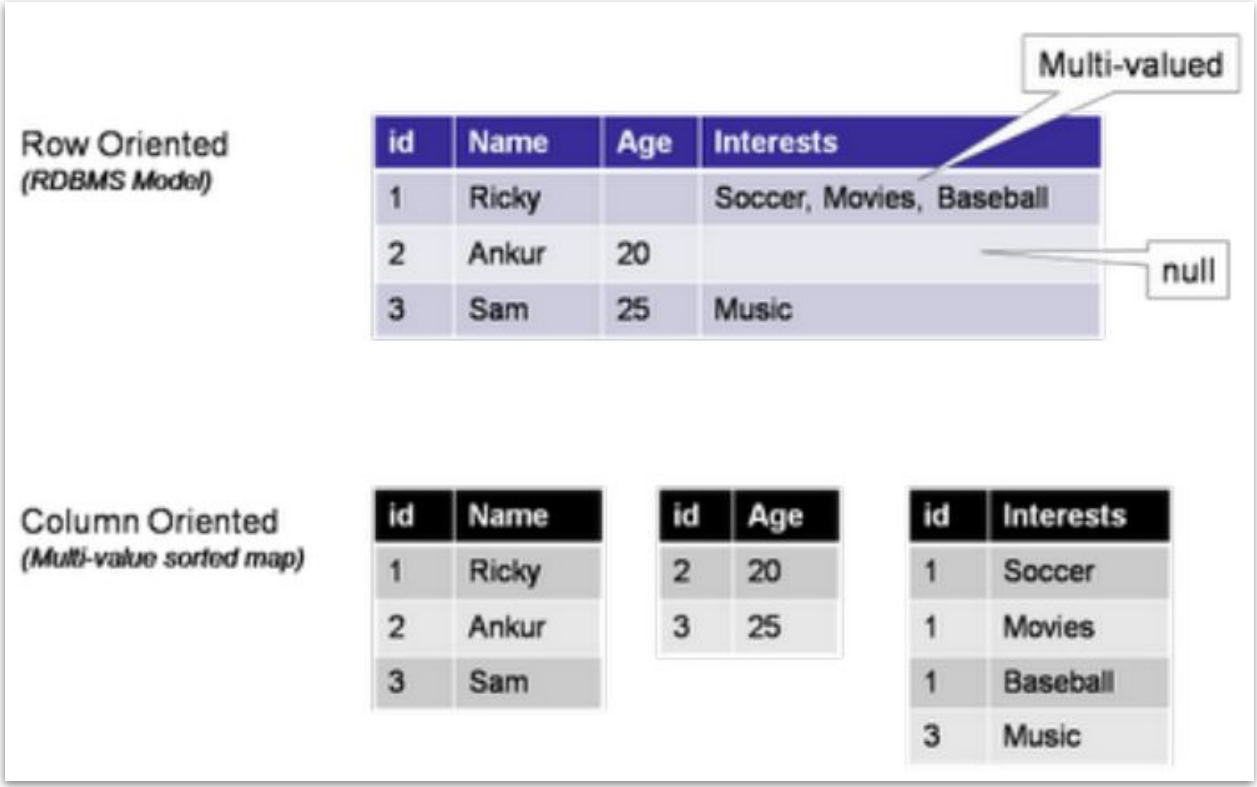
	S3	HDFS	S3 vs HDFS
Elasticity	Yes	No	S3 is more elastic
Cost/TB/month	$23	$206	10X
Availability	99.99%	99.9% (estimated)	10X
Durability	99.999999999%	99.9999% (estimated)	10X+
Transactional writes	Yes with DBIO	Yes	Comparable

https://databricks.com/blog/2017/05/31/top-5-reasons-for-choosing-s3-over-hdfs.html

A comparison of the complete package

- At a minimum, the complete package includes
  - Storage cost
  - Compute cost
  - Network cost
  - Operations cost (cost to operate daily and time to first query)
  - People cost
- For simplicity, let us consider only the compute and storage costs of building a big data cluster on a public cloud and on prem
  - Number of servers: 800, equally divided into two data centers
  - Server details: 512GB memory, 45TB disk, 2 CPUs with 20 cores each
  - Cluster details: 400TB memory, 36PB disk, 32,000 CPU cores
  - On prem: 800 * $12,500 = $10M with 3-year amortization
  - Public cloud: depends but $1M or more per month
- For large big data clusters, public cloud seems more expensive but it has its own advantages, as covered before

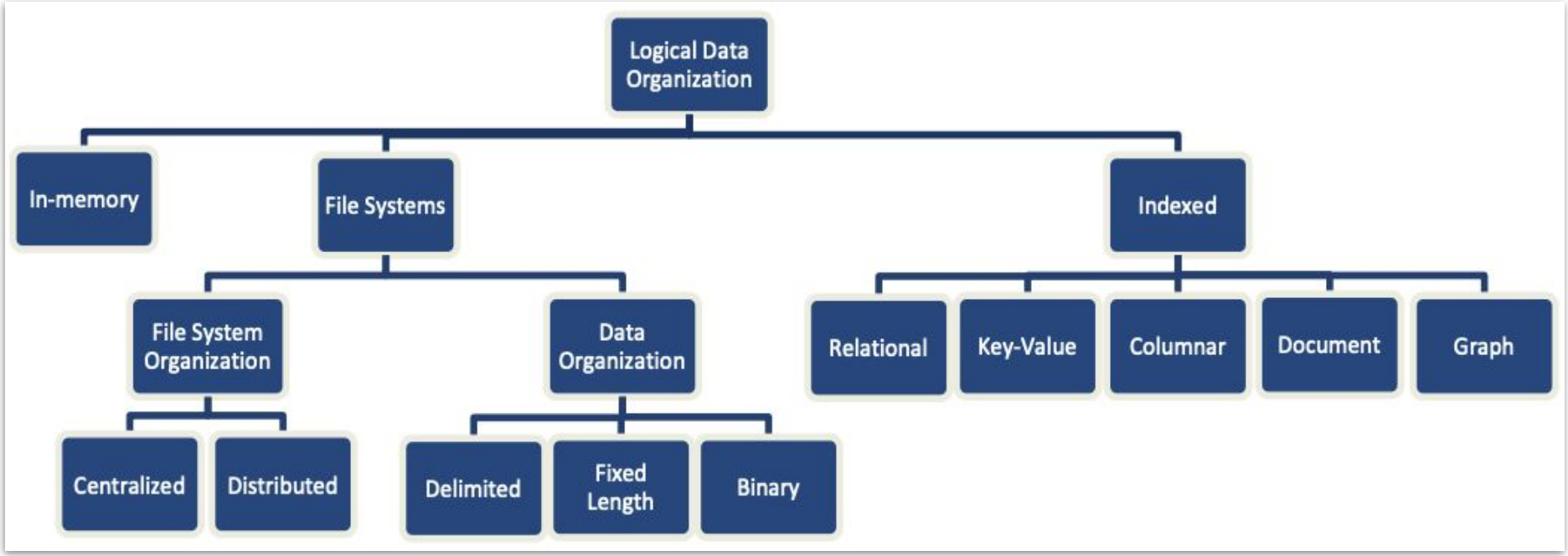
Refresher: Row vs column storage format



- Row format
  - Good for reading all fields in a record
  - Good for updates
  - Good for more transactional needs
- Columnar format
  - Good for reading few fields in a scan
  - Good for compression
  - Not ideal for updates
  - Good for more analytics needs

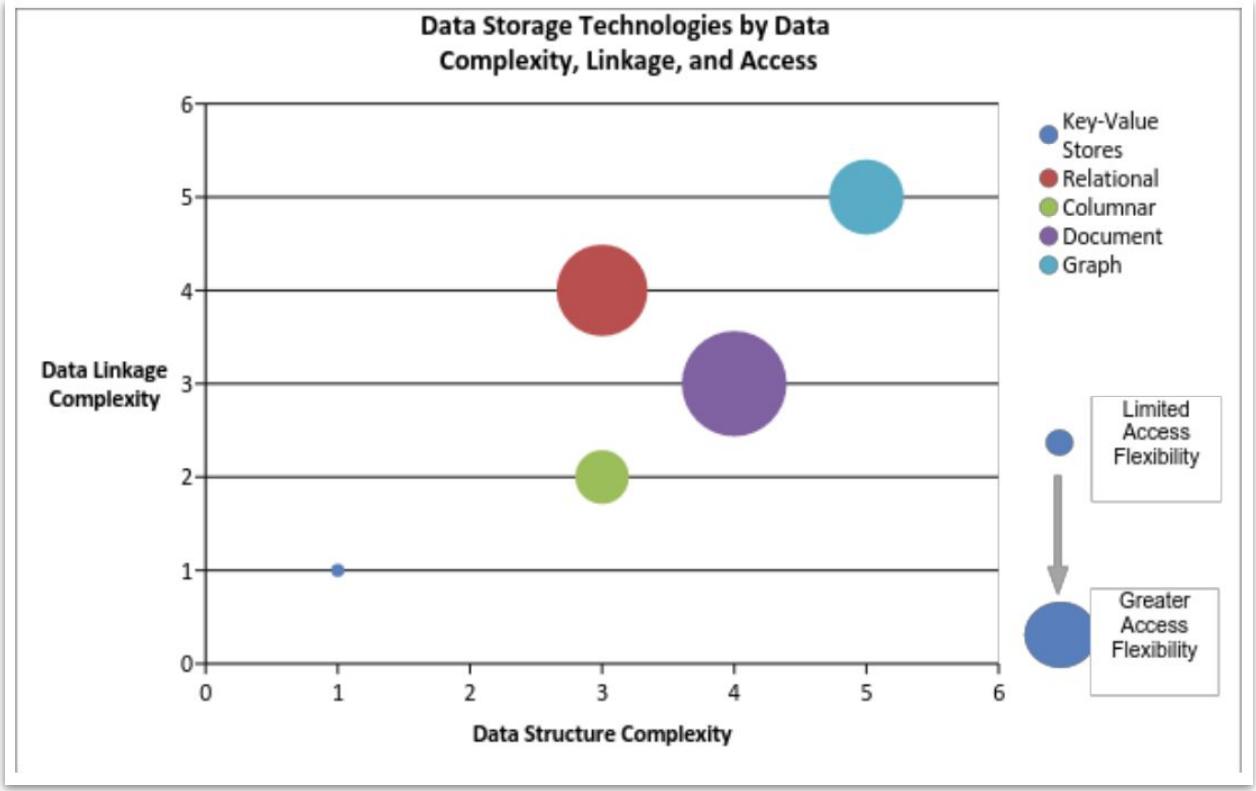
https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-6r1.pdf

Data organization choices



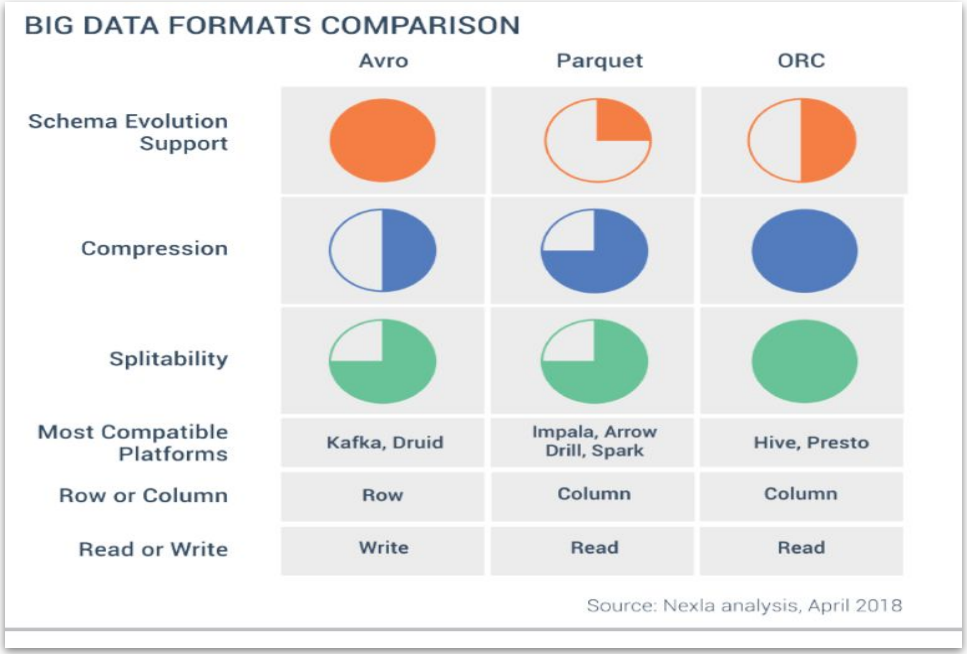
https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-6r1.pdf

Data organization comparison



https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-6r1.pdf

Data formats



- Row format
  - Apache Avro
- Columnar
  - Apache Arrow
  - Apache Parquet
  - Apache ORC
- Columnar and in-memory optimization
  - Apache Arrow

Encryption

- Data encryption possibilities
  - Data at rest: Inactive data stored physically in persistent storage
  - Data in process: Active data in memory manipulated by an application
  - Data in transit: Active data traveling between servers or systems
- Key management options
  - Customers manage their own keys
  - You manage all keys
- Key management tools
  - Use the relevant public cloud solutions
  - Use the solution provided by the open source distribution partner
- Processing over encrypted data
  - Decrypt using keys, managed by the application
  - Run processing over encrypted data without decrypting
    - Limited scope
    - Research solutions

Data transformation

Section 5 of 10

Four kinds of data

Raw data

- Data or logs ‘as is’
- “Source of truth”
- Data source for tables, profiles, and graphs
- Archival
- Transformed for unified formats
- Ex: All event logs

Tables

- For focused, fast access
- Ex (advertising): Clicks, actions
- Ex (retail): Sales

Profiles

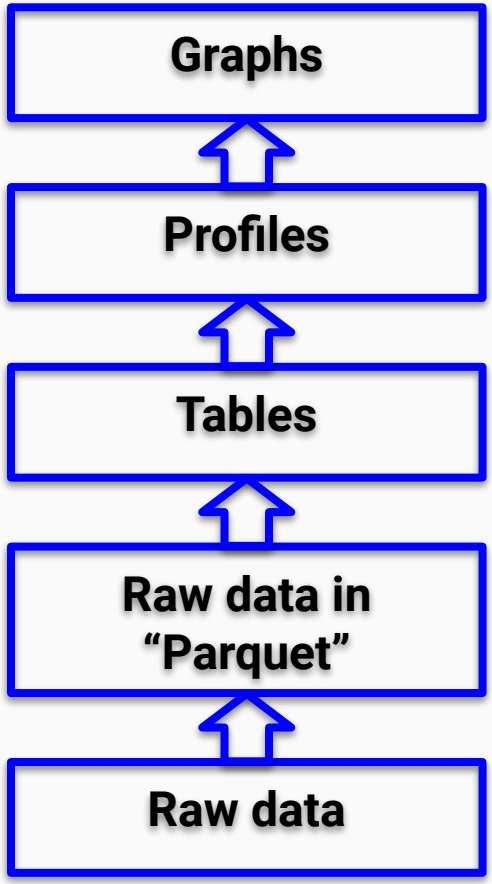
- For each key object in the space
- 360-degree view of each object
- For fast access to relevant fields
- Having given and derived fields
- Ex (advertising): Anonymized users
- Ex (retail): Customer, product, supplier

(Knowledge) Graphs

- Modeling relationships between profiles
- Having given and derived fields for both nodes and edges
- Reveals new observations
- Ex (retail): Customers to products to suppliers to ...

Pipelines to create four kinds of data

- Pull from source systems of raw data into the platform ‘as is’
- Transform raw data into a unified format for unified and easy access
- Stream new data into existing tables, or create new tables if needed
- Stream new data into existing profiles, or create new profiles if needed
- Drive new fields or enhance field values for profiles
- Stream changes into graphs, or create new graphs if needed
- Enhance properties of graph nodes and edges
- Push data and insights into destination systems



Other types of data pipelines

- ETL
- Reporting
- Analytics
- Aggregates for serving DB or tools
- ML feature engineering
- ML model building
- … and other application specific pipelines

Compute & Access layers

Section 6 and 7 of 10

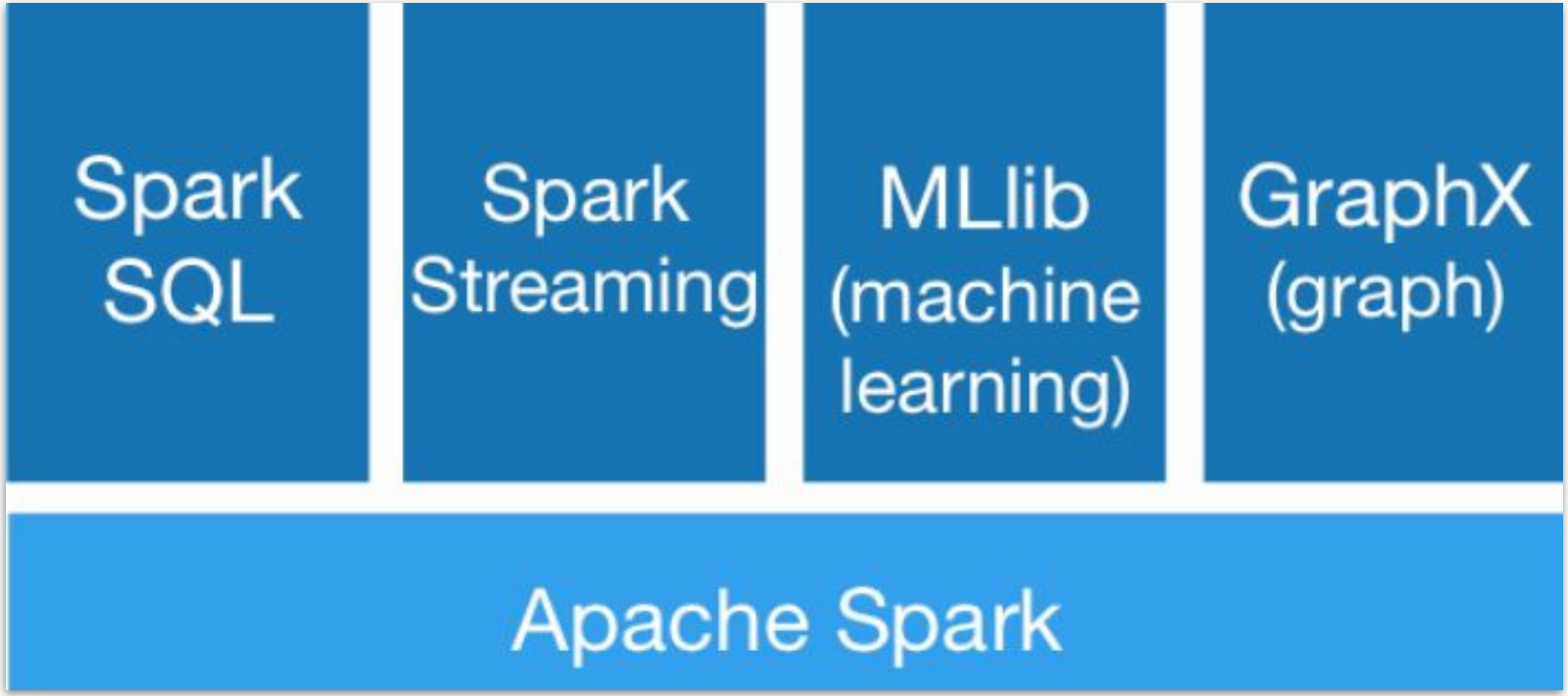
Custom applications

- Multiple languages and tools available for creating custom applications
- Java and later Pig were more common in the past
  - There is probably a large number of applications in these languages now that work just fine.
  - They were built on top of Apache Hadoop's MapReduce functionality.
  - If large, the cost of migrating to a more popular language or tool can be costly.
- Python is more common now
- Prototyping can also be done in SQL or Python.
- For input data flows, Apache Nifi can be used.

Applications via systems

- Apache Spark is the preferred choice right now.
  - Has support for Graphs, ML, SQL, and Streaming
  - Can use Python or Scala or SQL on Spark.
- Apache Hive is another choice for SQL interface.
- Prototyping can be done in SQL on Spark and can later move to Python or Scala for speed and ease of expressiveness.
- Notebooks via Python or R are also very popular for their ease of use and integrated approach in data science applications.
- Established tools like Matlab and SPSS also support Hadoop and Spark now.
- Lots of commercial tools with SQL-based access

Apache Spark



https://spark.apache.org/

Workflow management

- Many tools are available
  - Apache Airflow (from Airbnb)
  - Apache Oozie
  - Netflix Genie
  - Spotify Luigi
  - Linkedin's Azkaban
- Public clouds have their own tools.
- Each of these tools uses a DAG, with each node representing a job and each directed edge representing precedence or dependence between jobs.
- These tools are compared based on their expressiveness, input language, the ease of use with or without a GUI, reliability, etc.
- Apache Airflow seems to be the preferred choice now.

A comparison of workflow management tools

	Airflow	Azkaban	Conductor	Oozie	Step Functions
Owner	Apache (previously Airbnb)	LinkedIn	Netflix	Apache	Amazon
Community	Very Active	Somewhat active	Active	Active	N/A
History	4 years	7 years	1.5 years	8 years	1.5 years
Main Purpose	General Purpose Batch Processing	Hadoop Job Scheduling	Microservice orchestration	Hadoop Job Scheduling	General Purpose Workflow Processing
Flow Definition	Python	Custom DSL	JSON	XML	JSON
Support for single node	Yes	Yes	Yes	Yes	N/A
Quick demo setup	Yes	Yes	Yes	No	N/A
Support for HA	Yes	Yes	Yes	Yes	Yes
Single Point of Failure	Yes (Single scheduler)	Yes (Single web and scheduler combined node)	No	No	No
HA Extra Requirement	Celery/Dask/Mesos + Load Balancer + DB	DB	Load Balancer (web nodes) + DB	Load Balancer (web nodes) + DB + Zookeeper	Native
Cron Job	Yes	Yes	No	Yes	Yes
Execution Model	Push	Push	Poll	Poll	Unknown
Rest API Trigger	Yes	Yes	Yes	Yes	Yes
Parameterized Execution	Yes	Yes	Yes	Yes	Yes
Trigger by External Event	Yes	No	No	Yes	Yes
Native Waiting Task Support	Yes	No	Yes (external signal required)	No	Yes
Backfilling support	Yes	No	No	Yes	No
Native Web Authentication	LDAP/Password	XML Password	No	Kerberos	N/A (AWS login)
Monitoring	Yes	Limited	Limited	Yes	Limited
Scalability	Depending on executor setup	Good	Very Good	Very Good	Very Good

https://medium.com/@xunnan.xu/workflow-processing-engine-overview-2018-airflow-vs-azkaban-vs-conductor-vs-oozie-vs-amazon-step-90affc54d53b

Metadata and access management

- Many tools
  - Apache Atlas
  - Apache Ranger
  - Apache Knox
  - Netflix Metacat
  - Ab Initio (commercial)
  - Alation (commercial)
  - Collobra (commercial)
- Many other tools have their internal management functionality
  - E.g., Rockset

Output layer

Section 8 of 10

Refresher: Types of databases

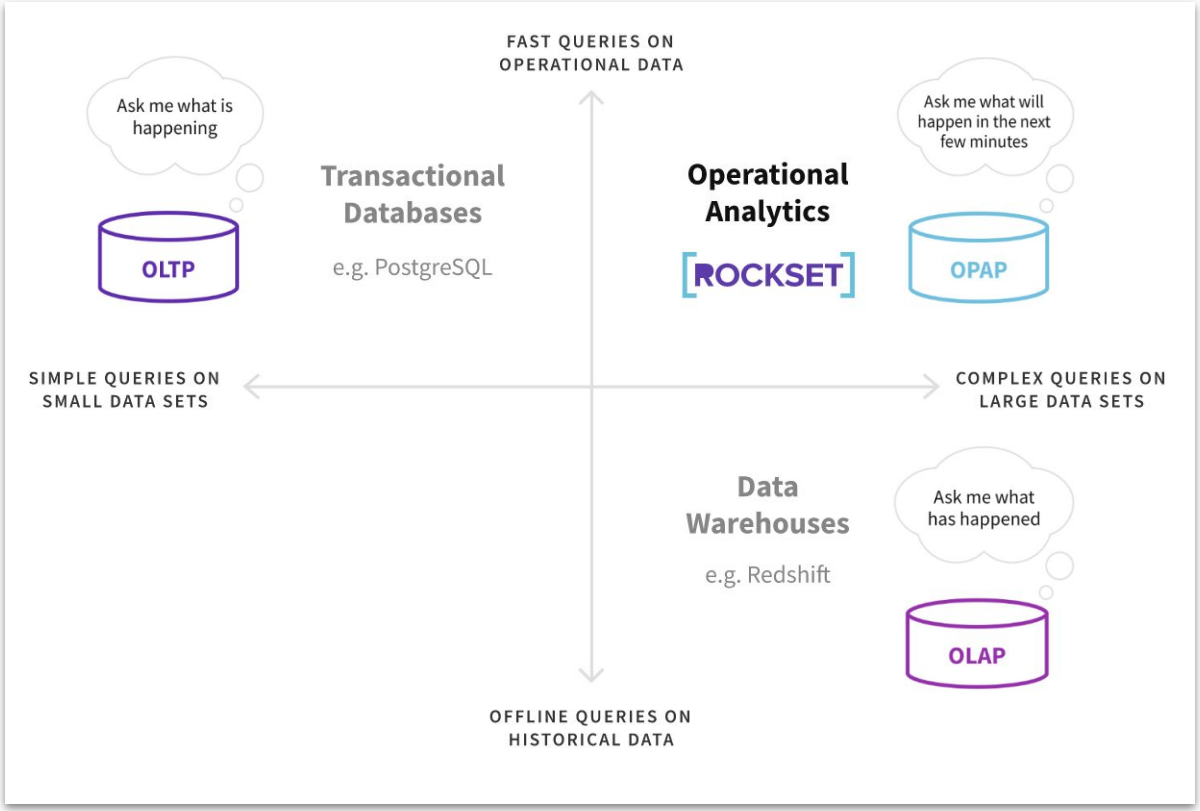
Transactional applications

Analytics applications

Operational applications

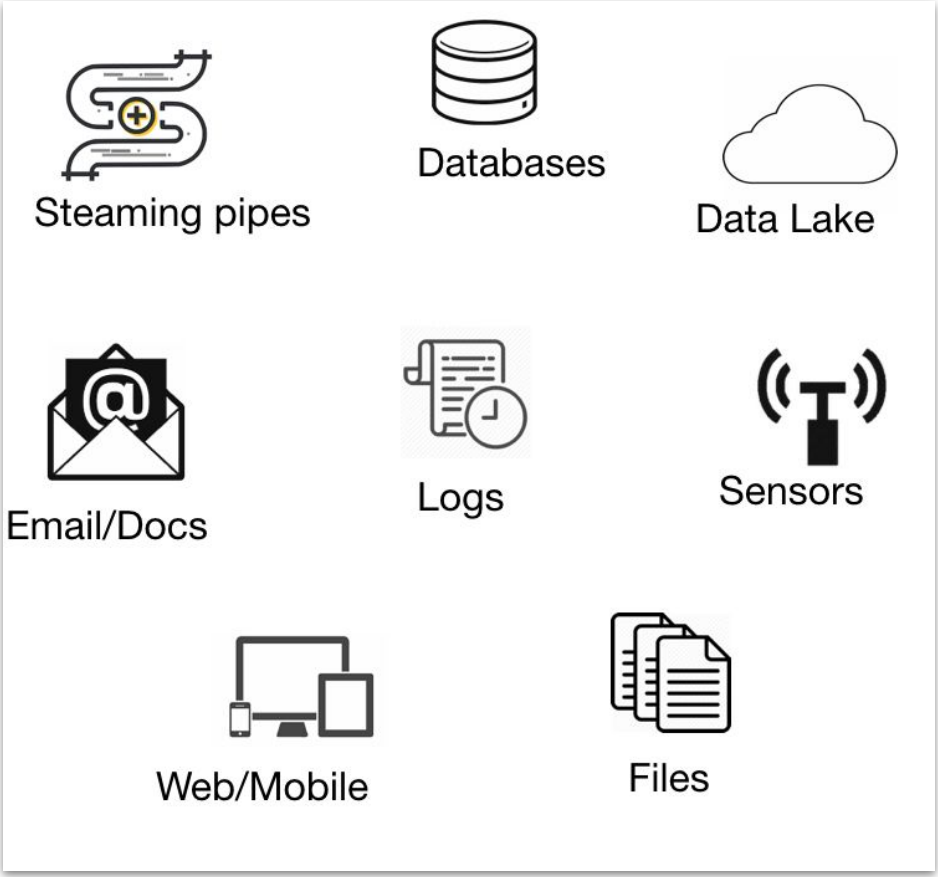
https://rockset.com/blog/operational-analytics-what-every-software-engineer-should-know/

Refresher: Types of databases



Output choices

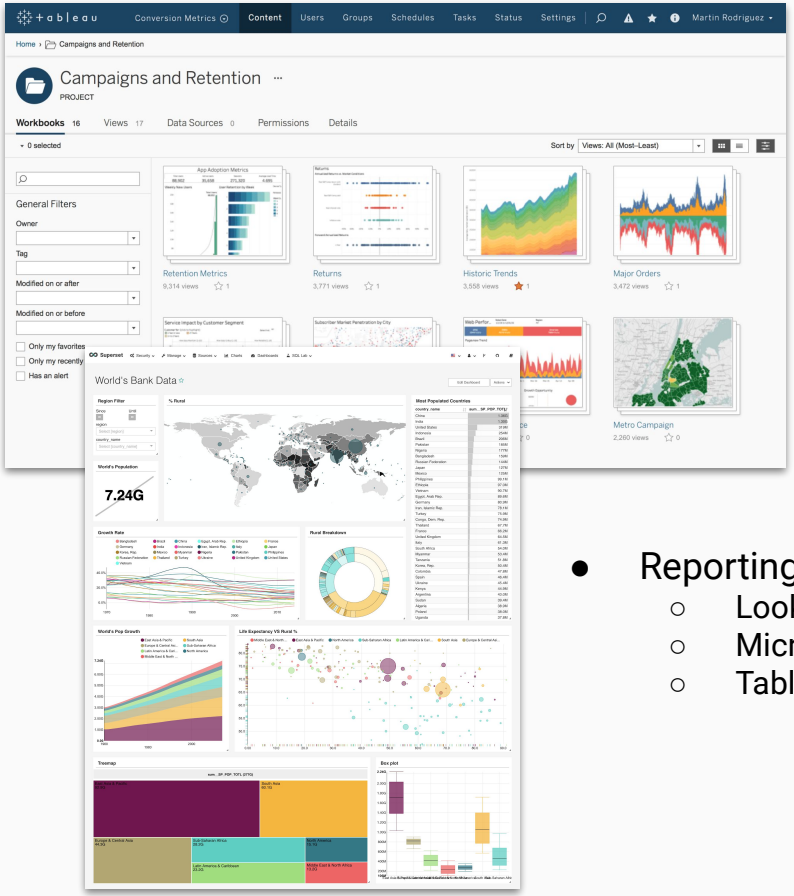
- Keep the output in the platform
- Push the output to a serving db
  - Reporting
  - Timeseries
  - Key/value
  - Search and/or document
  - Graph
- Push the output to another system
  - Another data platform
  - Streaming system
  - Another destination, possibly external



Serving DB choices

- Two key use cases
  - Reporting
  - Operational
- Reporting use case
  - Usually for analytics
  - More periodic, slower pace
  - Connected to more static dashboards
- Operational use case
  - Usually for operations
  - More real-time
  - Connected to more dynamic / live dashboards
- Data stores for reporting
  - Postgres, MariaDB, Druid
  - Commercial solutions: Oracle, Teradata.
  - Public cloud solutions: Internal
- Data stores for operational
  - Tsdb
  - Commercial solutions: Rockset, Elastic Search, Redis, InfluxDB
  - Public cloud solutions: Internal

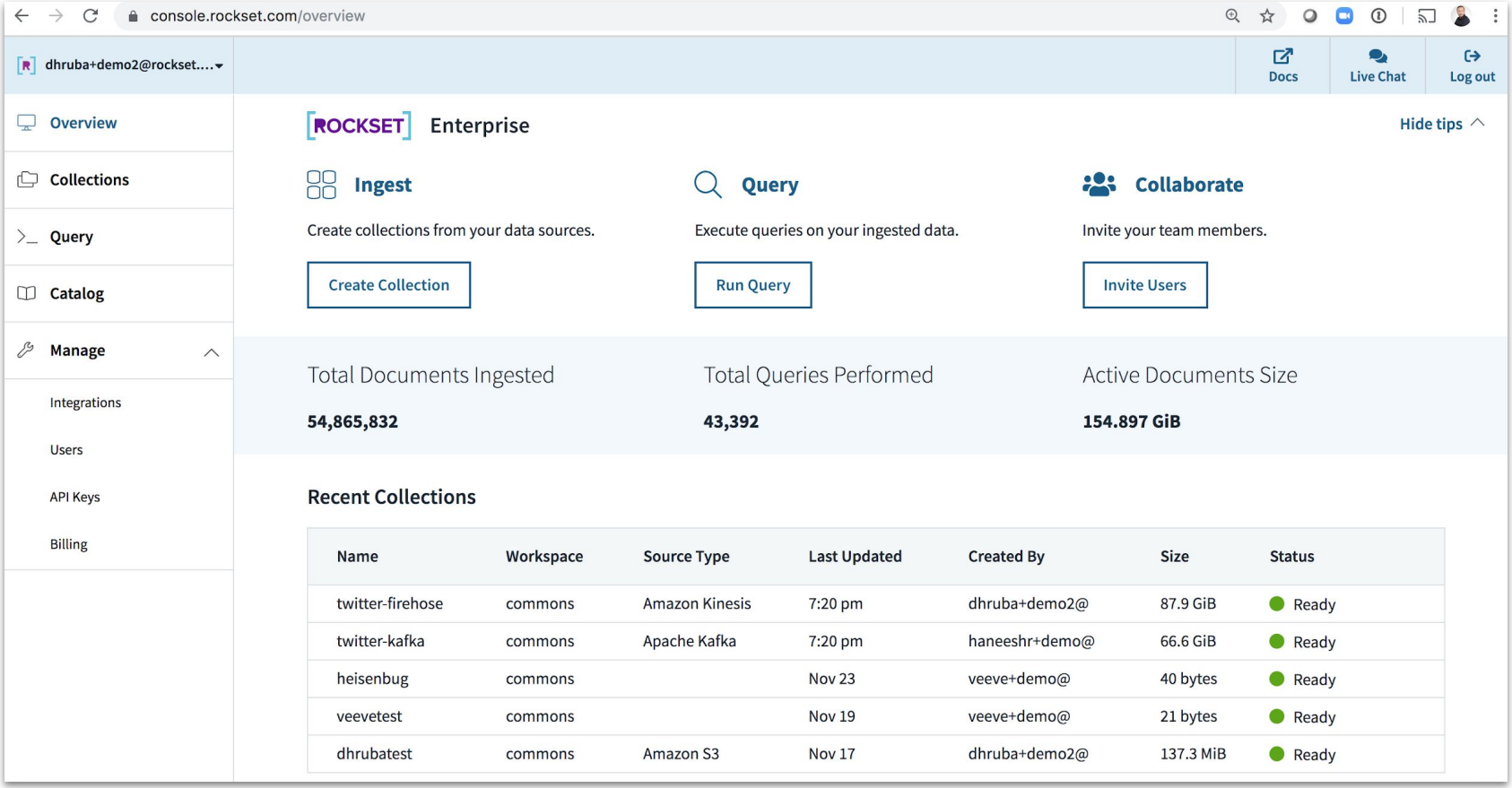
Business intelligence interfaces



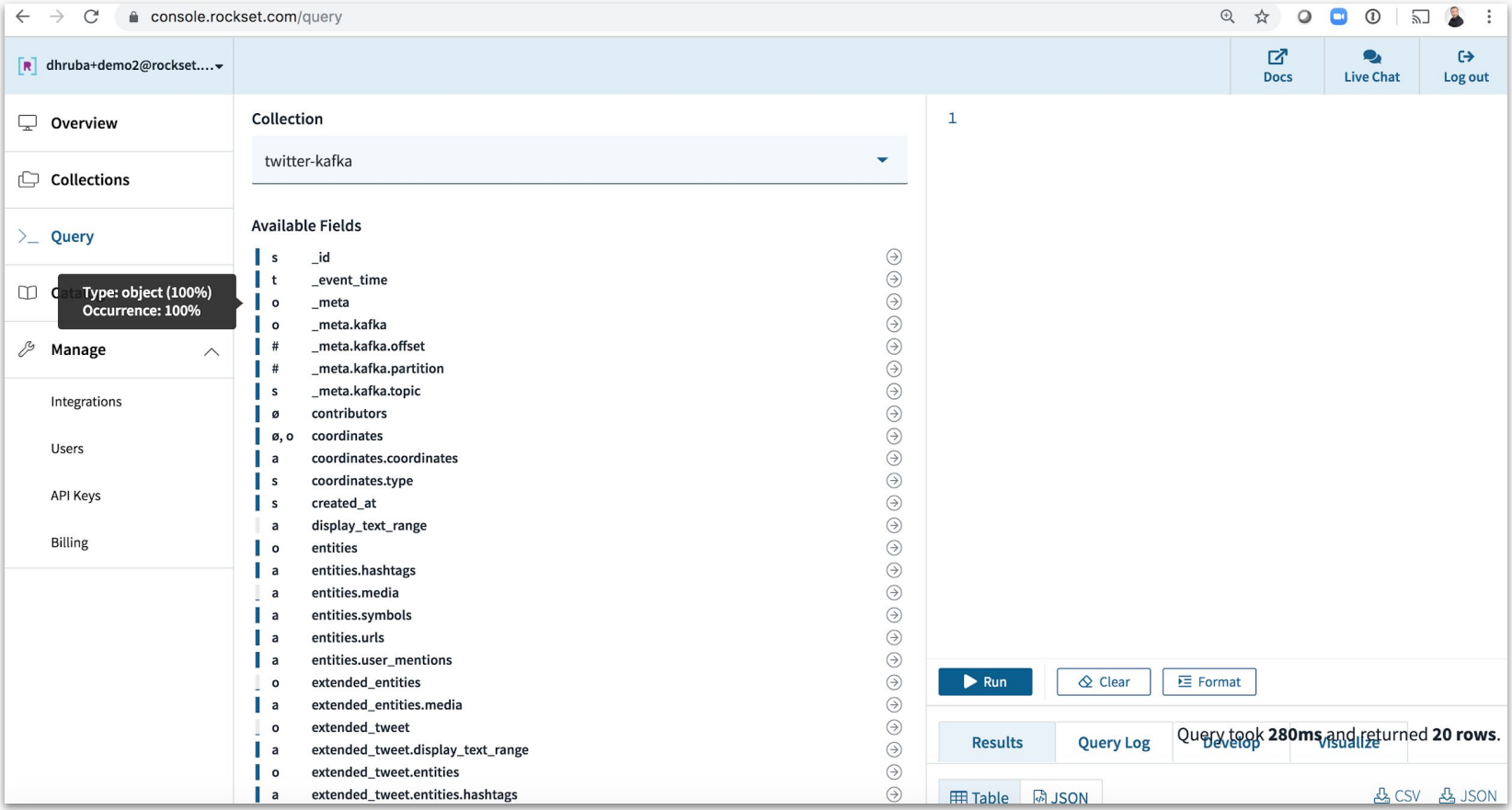
- Reporting
  - Looker
  - Microstrategy
  - Tableau

- Operational
  - Grafana
  - Apache superset

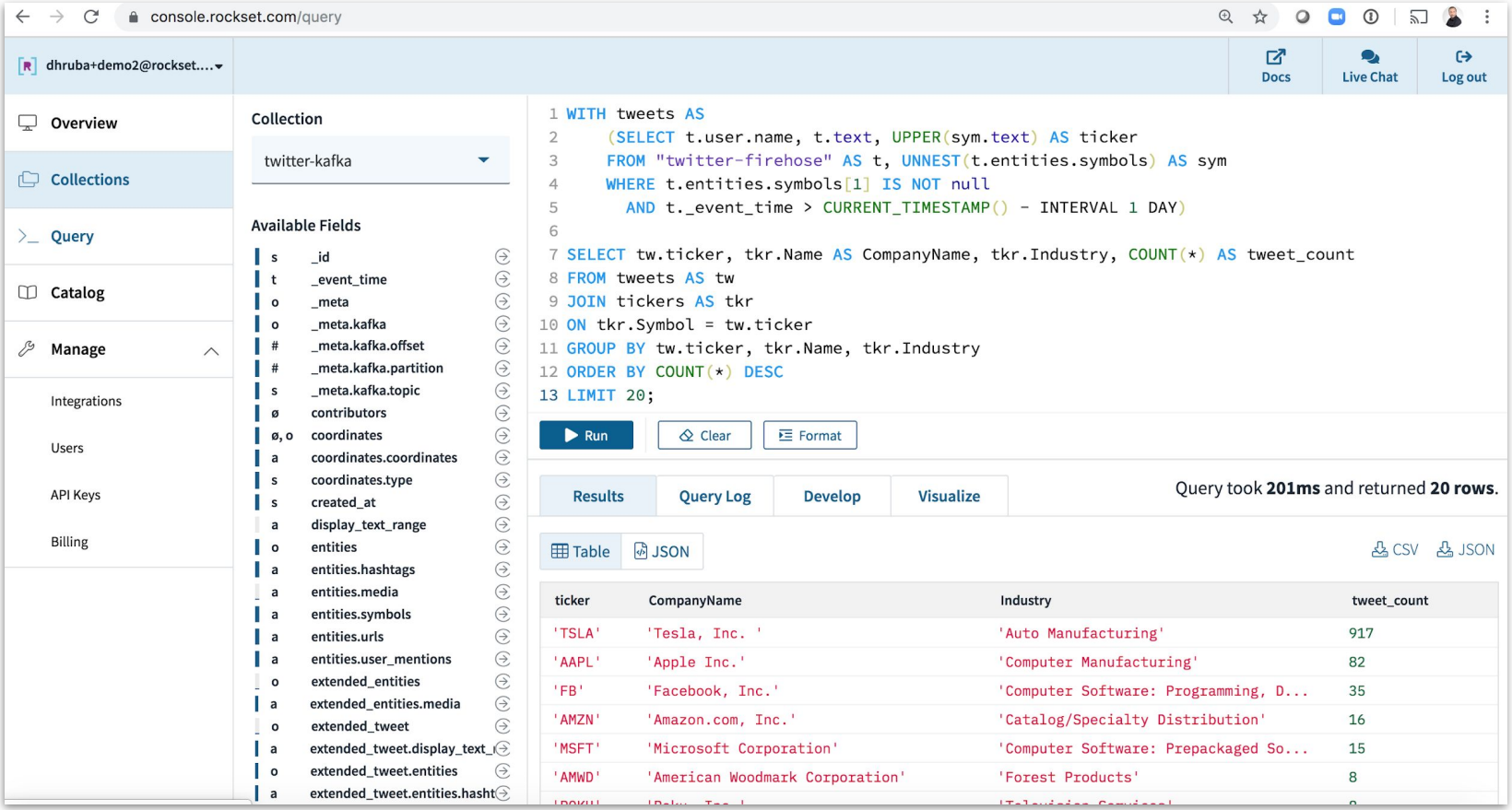
Serving DB interface example: Overview



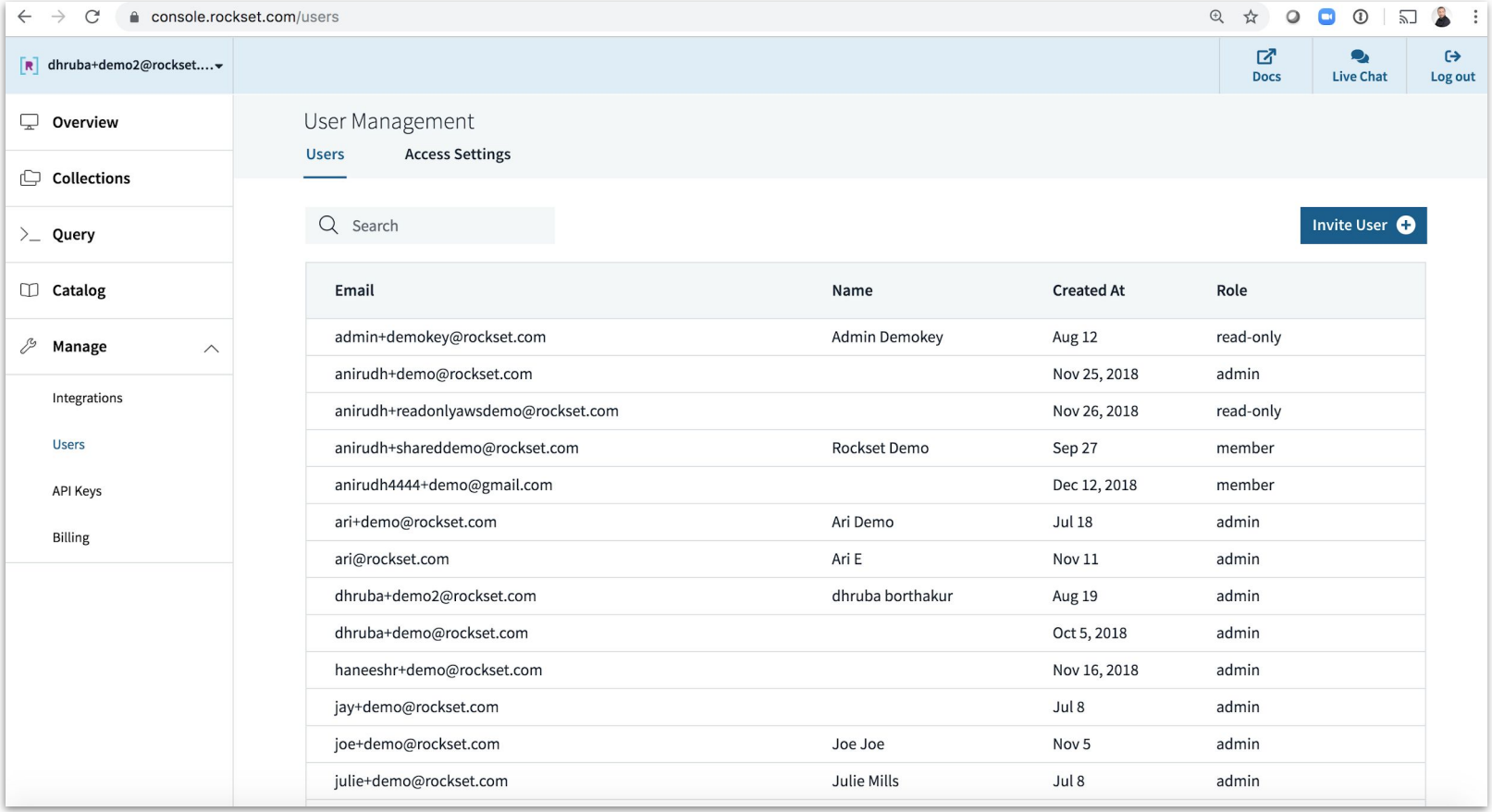
Serving DB interface example: Metadata



Serving DB interface example: Query



Serving DB interface example: Management



Management and operations

Section 9 of 10

Service Level Indicators, Objectives, and Agreements

- Data platforms usually start service without any guarantees.
- Users also do not expect any guarantees as they usually experiment or prototype.
- However, within six or so months, as more production jobs are moved to the data platform, service level guarantees will be demanded.
- In our experience, a sign of success with data platformization is that more and more users start complaining about such service level experiences as the speed of job execution, the number of users, reliability, and availability of data sources.
- Service level indicators are common for all users.
  - Every user expects reliable operation.
- Service level objectives may be user or task dependent.
  - One user may require her job to execute once a day, another once every hour.

Service Level Indicators (SLIs) to guarantee

- Latency
- Frequency
- Data correctness, exactly-once semantics globally
- Availability
- Durability
- Reliability
- Mean time to resolve (MTTR) incidents
- Mean time to detect (MTTD) incidents
- Mean time to communicate incidents and outages
- Change management

- Resource usage, cost
- Ease of development
- Ease of deployment
- Ease of experimentation
- Ease of operation
- Ease of metadata management
  - for data, jobs, models, and features
- Ease of data ingress, data egress
- Availability of tools for access and insights
- Number of users supported

Usually the durability objective for data is far more stringent than the availability of the platform.

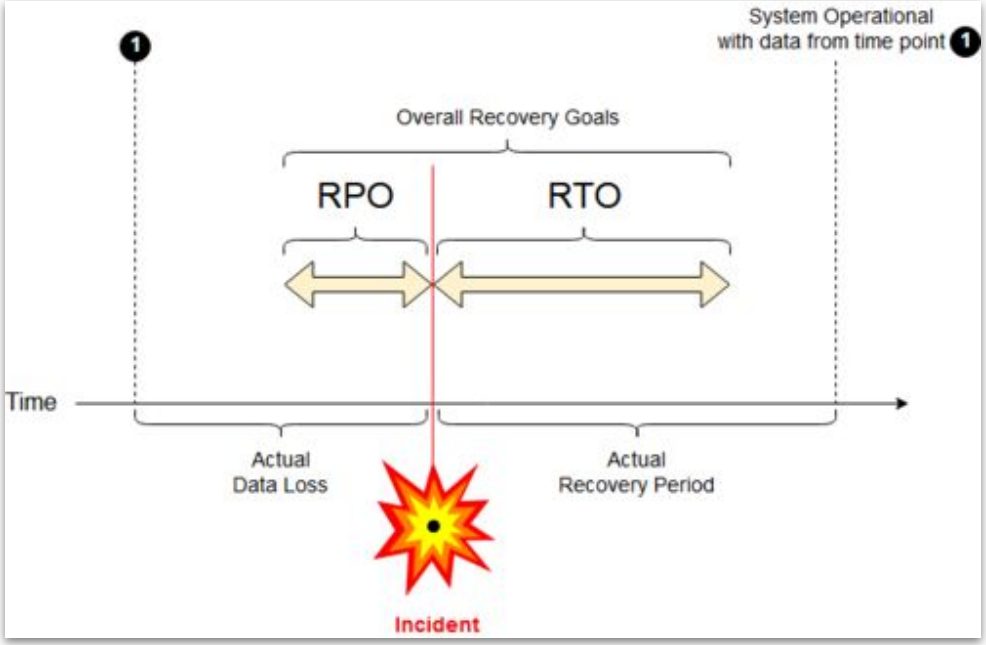
In other words, the platform may be down but should not lose data.

Multi-cluster management

- Need for multiple clusters
  - Different sets of users or applications, e.g., financial data with insiders vs not
  - Disaster recovery
  - Load reduction or balancing
  - Cluster size limitations due to, e.g., Hadoop software scaling limitations
- Explanation using the disaster recovery example: Given two clusters A and B
  - The same data on both at any given time
    - How? Constantly replicate data from A to B and vice versa
    - For improved data consistency, ingest data to A and then replicate to B or vice versa
  - The same applications ready to go on both at any given time
    - How? Deploy to both continuously and test at regular intervals
  - Each is sized to support the desired traffic load if the other goes down
  - Due to the cost of these clusters, it is not recommended to run them as active-passive. Both can be active for different sets of applications and users while each is ready to take the load from the other in case of a disaster

Disaster recovery

- RPO: Recovery Point Objective
  - Time during which data got lost due to an incident
- RTO: Recovery Time Objective
  - Time to recovery from an incident
- RPO and RTO may depend on the cluster, the data, the applications, etc.
- One way to minimize RPO
  - If raw data comes from sources outside the data platform, have these sources to keep a history longer than RPO
- RTO will include multiple activities
  - Ingesting the “lost” data from the sources
  - Starting the applications
  - Running the applications to produce the “lost” output
  - Running the applications to process the current incoming data



https://en.wikipedia.org/wiki/Disaster_recovery

Operations

- Users just want to focus on getting their reports and insights
- Need full operations support to ensure users are happy
- As a data platform gets more production usage and dependence, the expectations from and pressures on operations team will increase significantly so start early to build the necessary operations muscles
- Need an operations team just for the data platform
  - If the team is small and/or inexperienced, also get maintenance and service deals from vendors for hardware and software support
- Make sure the data, the pipelines, the applications are owned by non-operations team; the operations team needs to focus on ensuring that the platform runs well as an enabler for its users
- One sign of success of a data platform: Users complain about limited data platform resources, how long their applications wait in queues, how long their applications take time to run, etc.

The rest of management concerns

Other concerns

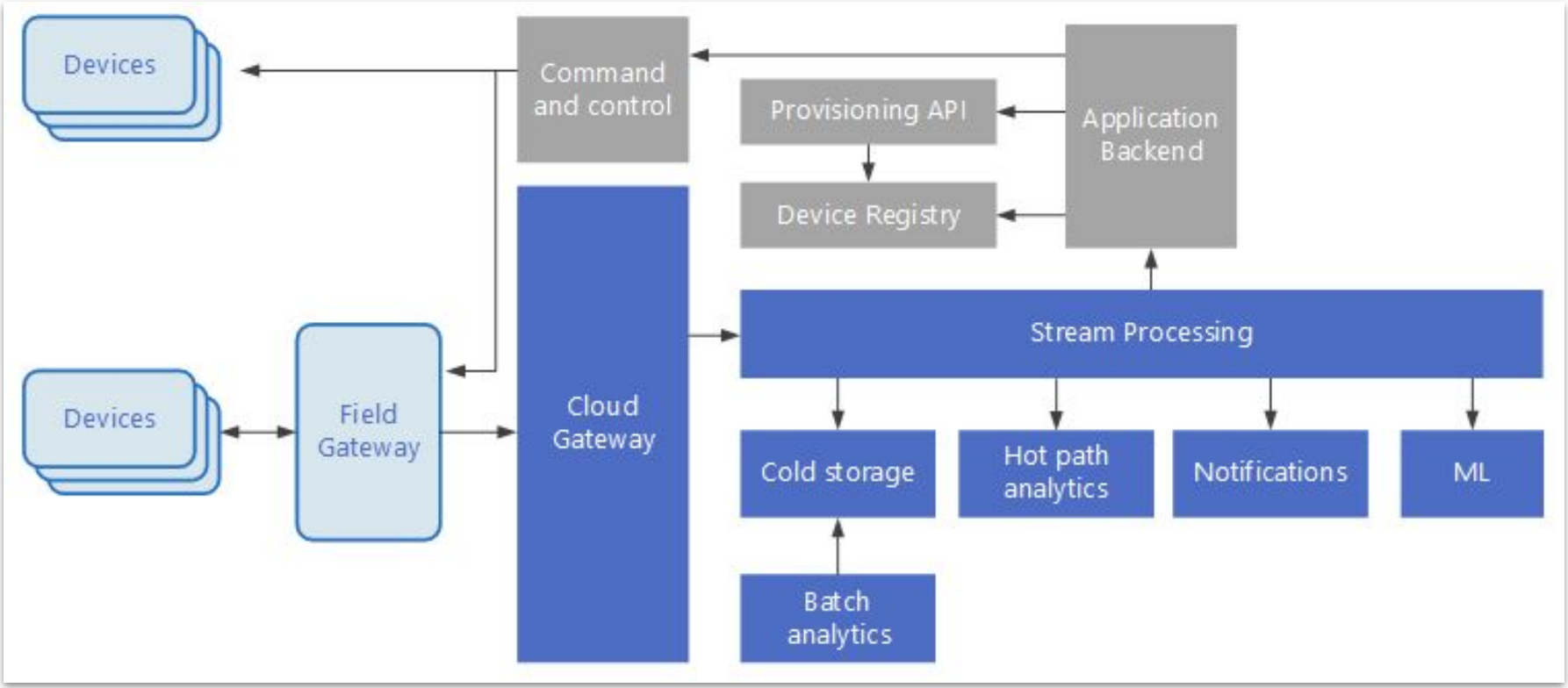
- Data governance
- Security
- Data locality
- Privacy
- GDPR

We will not cover these in this tutorial due to time and focus limitations.

Example: Machine Learning

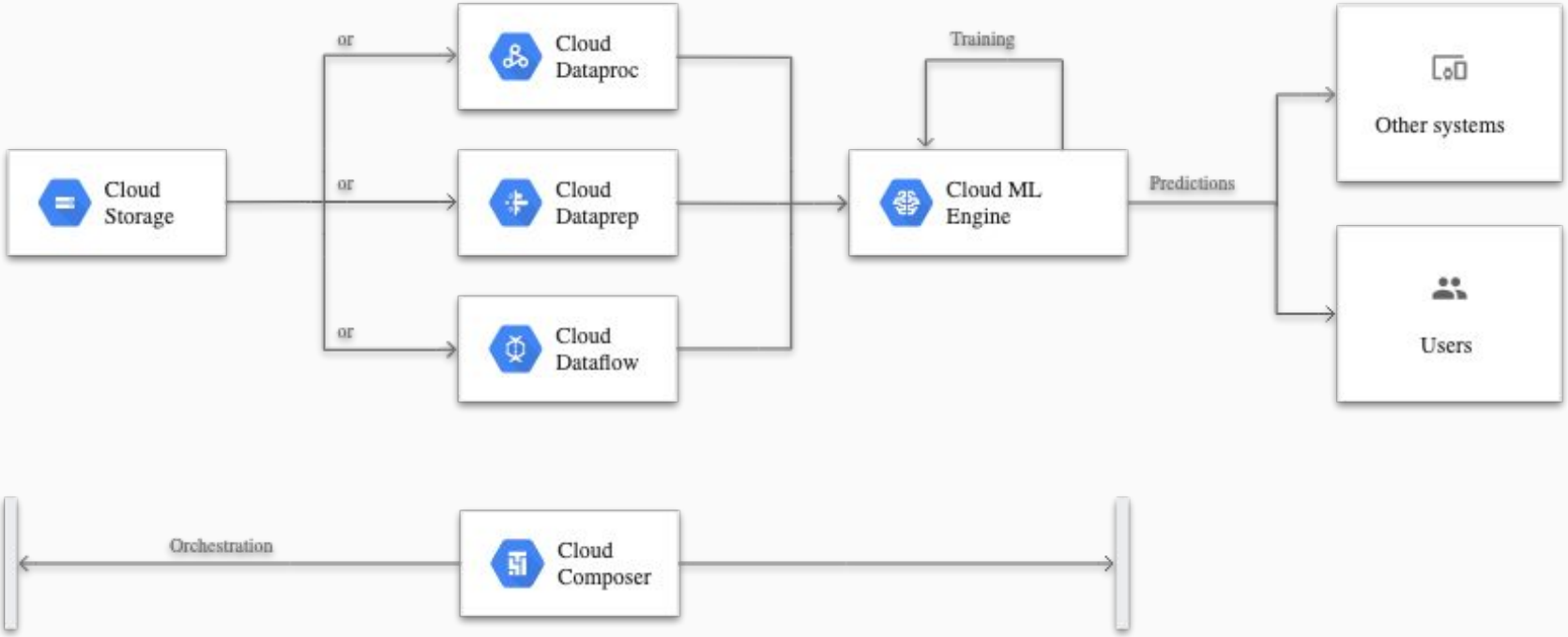
Section 10 of 10

Microsoft Cloud: Big data architecture for IoT



https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

Google Cloud: Machine learning processing



https://cloud.google.com/solutions/build-a-data-lake-on-gcp

Apple: Machine learning processing

Solution	Integrated Data System	Model Training	Experimentation	Evaluation	Deployment
SageMaker	No (Bring your own data store: *e.g.*, S3, EBS, EFS)	Yes	Yes	Yes	Yes
Azure ML	No (Bring your own data store)	Yes	Yes	Yes	Yes
Cloud AI	No (Bring your own data store: *e.g.*, Google Cloud Storage)	Yes	Yes	Yes	Yes
TensorFlow	N/A	Yes	No	No	Yes
Keras	N/A	Yes	No	No	No
PyTorch	N/A	Yes	No	No	No
Michelangelo	No (HDFS data lake, Cassandra)	Yes	Yes	Yes	Yes
FBLearner Flow	N/A	No	Yes	No	Yes
Colaboratory	N/A	No	Yes	Yes	No
Databricks	N/A	No	Yes	Yes	No
MLdp	Yes	Integrated with major ML frameworks	Integrated with in-house execution clusters, as well as public elastic compute cloud	Integrated with an in-house solution	Integrated with an in-house solution

Table 1: Machine learning platform eco-System

https://dl.acm.org/citation.cfm?id=3314050

Apple: ML from conventional to integrated

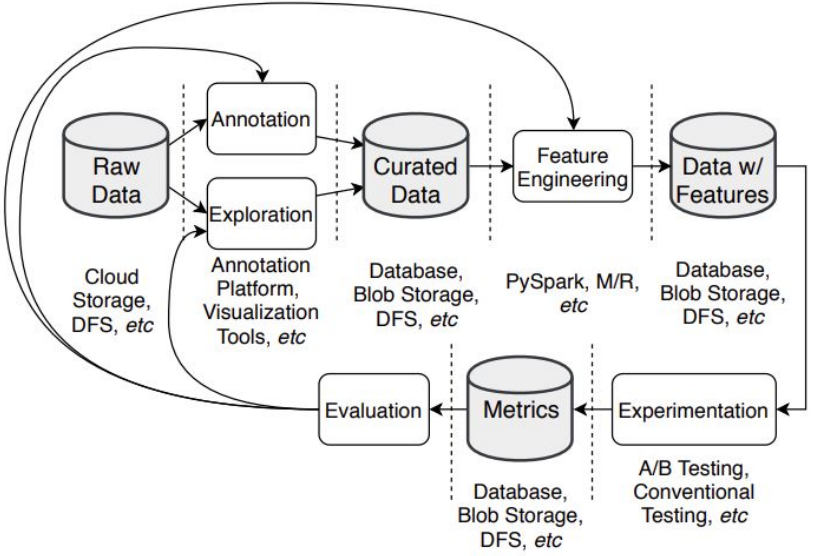


Figure 1: Conventional ML workflow and data silos.

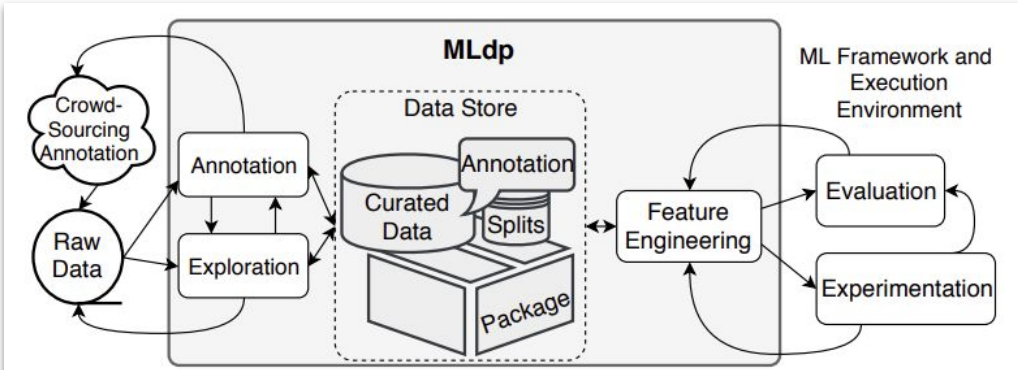


Figure 2: Integrated ML and data workflow: MLdp acts as data interface with data model, data access and life cycle management.

https://dl.acm.org/citation.cfm?id=3314050

Apple: ML data platform architecture

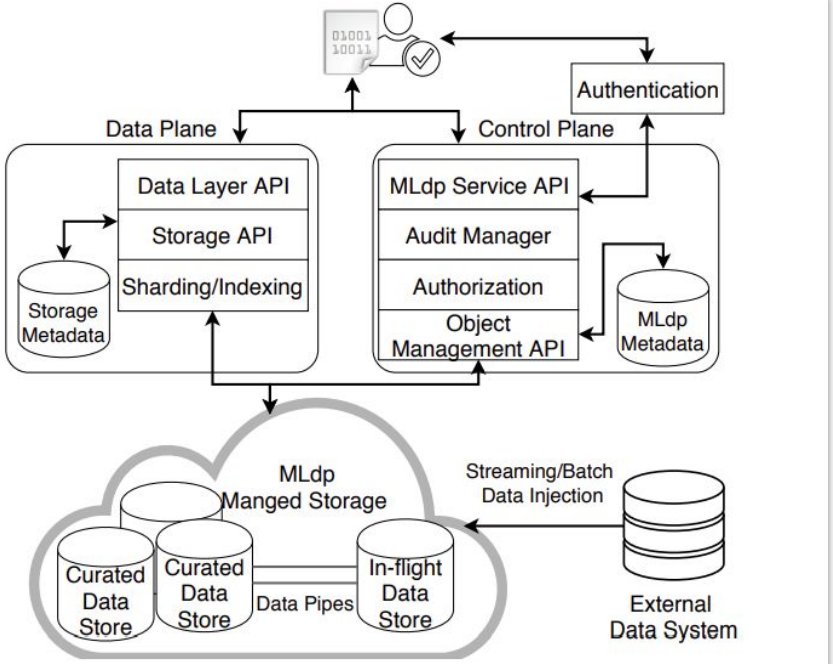


Figure 3: MLdp system architecture showing control plane and data plane and hybrid storage model.

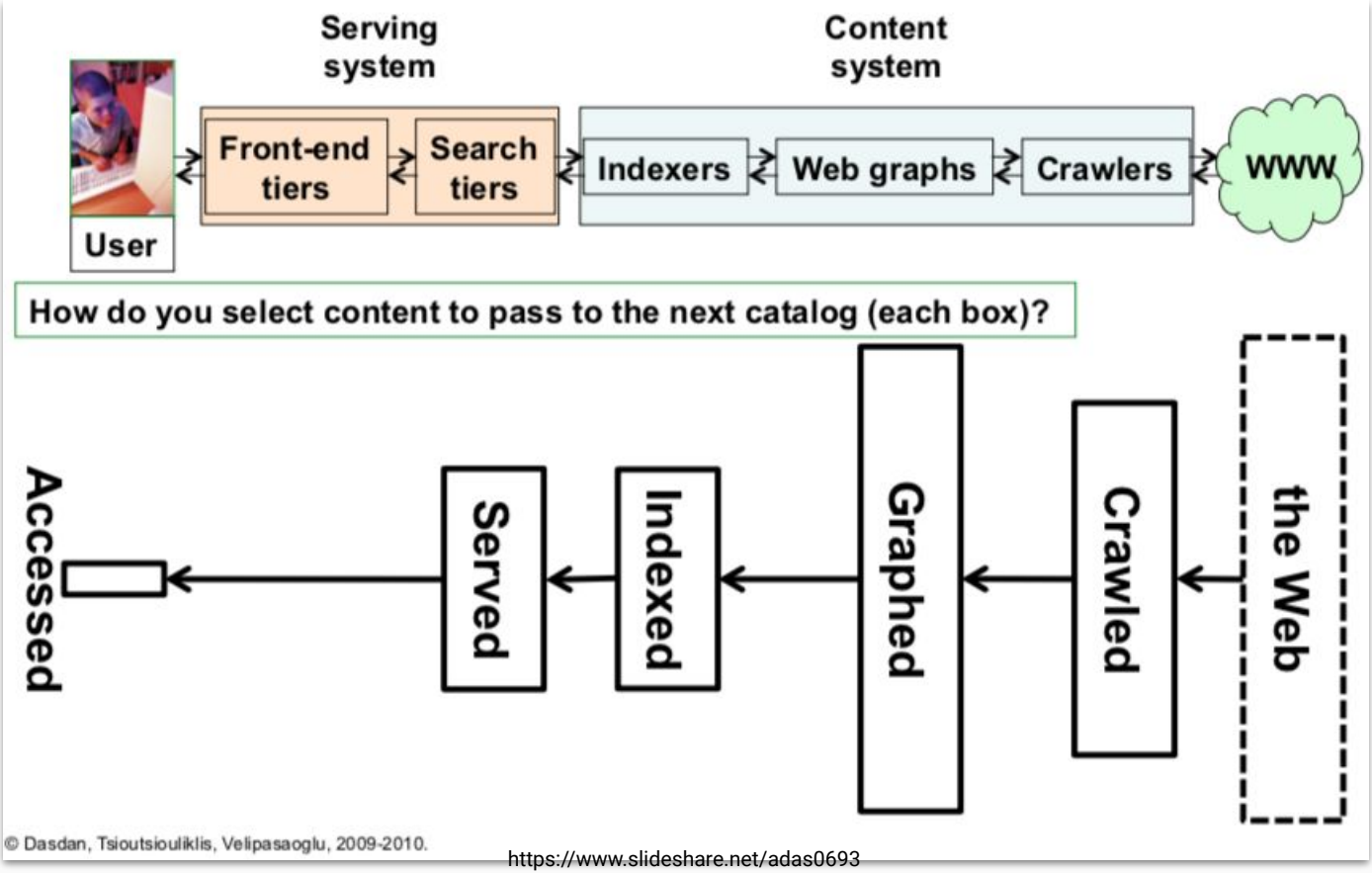
https://dl.acm.org/citation.cfm?id=3314050

Example:
Web Search Log Analysis

Section 10 of 10

The 1st production application on Hadoop, internally called “key site monitoring”, deployed in the summer of 2006 at Yahoo!

Refresher: Web Search Engine Architecture

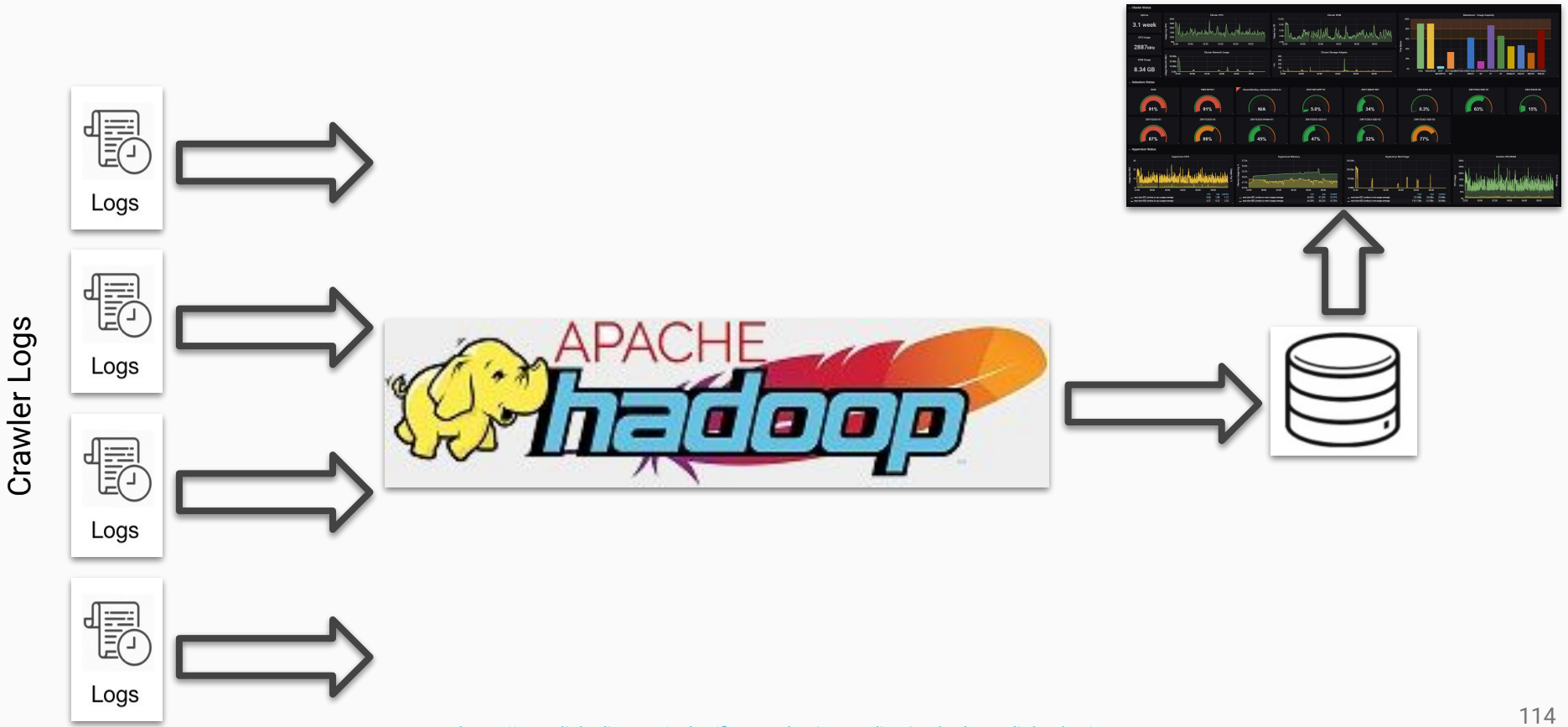


https://www.slideshare.net/adas0693

Crawler Log Analysis: Key Site Monitoring

- A web search engine gets content by crawling the web sites on the WWW.
- Key sites were the most important sites that our crawlers were configured to always crawl for these sites' quality content.
  - Key sites were derived using a PageRank-like algorithm on the web graph.
- Key site monitoring (KSM) was to ensure that these sites were always crawled satisfactorily every day.
- The (simplified) KSM process worked as follows:
  - Pull in a day's log files from a sample of crawler nodes into HDFS
  - Scan these log files for daily statistics for key sites in parallel using MapReduce on Hadoop
  - Add daily statistics to the existing time series for key sites on a DB outside Hadoop
  - Provide graphical visualization and alerting over these time series
- v1.0 vs v2.0
  - v1.0: Move **large** log files from a **sample of crawler nodes** into HDFS; scan them **in parallel on Hadoop**; send tiny stats data into monitoring DB
  - v2.0: Scan log files **in each crawler node in parallel** for daily stats; send **tiny** stats data into monitoring DB

Key Site Monitoring Architecture



https://www.linkedin.com/pulse/first-production-application-hadoop-ali-dasdan/

Example:
Online Real-Time Advertising

Section 10 of 10

Resources to learn the details: Systems

- How does online advertising work?
  - https://www.linkedin.com/pulse/how-does-online-real-time-advertising-work-ali-dasdan/
- Architecture of a generic online advertising platform
  - https://www.linkedin.com/pulse/architecture-generic-online-real-time-advertising-platform-ali-dasdan/
- From 0.5 Million to 2.5 Million: Efficiently Scaling up Real-Time Bidding
  - https://ieeexplore.ieee.org/document/7373421
  - http://www0.cs.ucl.ac.uk/staff/w.zhang/rtb-papers/turn-throatling.pdf
- Overview of Turn Data Management Platform for Digital Advertising
  - http://www.vldb.org/pvldb/vol6/p1138-elmeleegy.pdf
- A High Performance, Custom Data Warehouse on Top of MapReduce (similar to Hive)
  - https://www.vldb.org/pvldb/vldb2010/pvldb_vol3/I08.pdf
- Leveraging Materialized Views For Very Low-Latency Analytics Over High-Dimensional Web-Scale Data
  - http://www.vldb.org/pvldb/vol9/p1269-liu.pdf

Resources to learn the details: Data Science

- Estimating Conversion Rate in Display Advertising from Past Performance Data
  - https://dl.acm.org/citation.cfm?id=2339651
  - https://s.yimg.com/ge/labs/v2/uploads/kdd20122.pdf
- Online Model Evaluation in a Large-Scale Computational Advertising Platform
  - https://arxiv.org/abs/1508.07678
- Multi-Touch Attribution Based Budget Allocation in Online Advertising
  - https://arxiv.org/abs/1502.06657
- User Clustering in Online Advertising via Topic Models
  - https://arxiv.org/abs/1501.06595
- Scalable Audience Reach Estimation in Real-time Online Advertising
  - https://arxiv.org/abs/1305.3014
- Real Time Bid Optimization with Smooth Budget Delivery in Online Advertising
  - https://arxiv.org/abs/1305.3011

Final: Lessons learned

Lessons learned

1. Start small, build it, and they will come
2. Put all data in one place
3. Build a general platform
4. Do not overbuild as you will rebuild
5. Use your own infrastructure
6. Ship early and often
7. A/B test relentlessly
8. Be patient: Innovation takes time to get accepted

References

References (1 of 12)

https://www.ibm.com/cloud/garage/architectures/dataAnalyticsArchitecture/reference-architecture

https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-6r1.pdf

https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

https://www.oracle.com/assets/oracle-wp-big-data-refarch-2019930.pdf#targetText=The%20reference%20architecture%20is%20defined,best%20practices%20in%20the%20industry.

http://eprints.bournemouth.ac.uk/30020/1/SKIMA2016_paper_76.pdf

https://core.ac.uk/download/pdf/83943361.pdf

https://www.emc.com/collateral/white-paper/h12878-rsa-pivotal-security-big-data-reference-architecture-wp.pdf

https://docs.cloudera.com/documentation/other/reference-architecture.html

https://www.dellemc.com/resources/en-us/asset/white-papers/solutions/cdh-architecture-guide.pdf

https://h20195.www2.hpe.com/v2/GetPDF.aspx/4AA5-9086ENW.pdf

https://www.cisco.com/c/dam/m/en_sg/dc-innovation/assets/pdfs/cloudera-enterprise-data-lake-presentation.pdf

https://lenovopress.com/tips1329.pdf

https://eng.uber.com/uber-big-data-platform/

References (2 of 12)

https://eng.uber.com/managing-data-workflows-at-scale/

https://medium.com/netflix-techblog/hadoop-platform-as-a-service-in-the-cloud-c23f35f965e7

https://www.youtube.com/watch?v=CSDIThSwA7s

https://medium.com/netflix-techblog/genie-2-0-second-wish-granted-d888d79455c6

https://medium.com/netflix-techblog/evolving-the-netflix-data-platform-with-genie-3-598021604dda

https://medium.com/netflix-techblog/announcing-suro-backbone-of-netflixs-data-pipeline-5c660ca917b6

https://medium.com/netflix-techblog/how-data-inspires-building-a-scalable-resilient-and-secure-cloud-infrastructure-at-netflix-c14ea9f2d00c

https://slack.engineering/data-wrangling-at-slack-f2e0ff633b69

https://medium.com/airbnb-engineering/data/home

https://medium.com/airbnb-engineering/data-infrastructure-at-airbnb-8adfb34f169c

https://cloud.google.com/data-catalog/

https://landing.google.com/sre/workbook/chapters/data-processing/

https://cloud.google.com/solutions/build-a-data-lake-on-gcp

https://ai.google/research/pubs/pub37795

References (3 of 12)

https://ai.google/research/pubs/pub48449

https://ai.google/research/pubs/pub47600

https://ai.google/research/pubs/pub46178

https://ai.google/research/pubs/pub48388

https://ai.google/research/pubs/pub36256

https://ai.google/research/pubs/pub36257

https://ai.google/research/pubs/pub44915

https://ai.google/research/pubs/pub41318

https://ai.google/research/pubs/pub41344

https://ai.google/research/pubs/pub47224

https://ai.google/research/pubs/pub38125

https://ai.google/research/pubs/pub37217

https://ai.google/research/pubs/pub37252

References (4 of 12)

https://ai.google/research/pubs/pub47669

https://ai.google/research/pubs/pub47828

https://ai.google/research/pubs/pub45394

https://ai.google/research/pubs/pub44686

https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/42851.pdf

https://ai.google/research/pubs/pub43287

https://ai.google/research/pubs/pub43462

https://ai.google/research/pubs/pub43985

https://ai.google/research/pubs/pub40465

https://arxiv.org/pdf/1905.12133.pdf

https://dl.acm.org/citation.cfm?id=3314050

https://blog.twitter.com/engineering/en_us/topics/infrastructure/2017/the-infrastructure-behind-twitter-scale.html

https://blog.twitter.com/engineering/en_us/topics/infrastructure/2019/partly-cloudy-architecture.html

References (5 of 12)

https://blog.twitter.com/engineering/en_us/topics/infrastructure/2019/democratizing-data-analysis-with-google-bigquery.html

https://blog.twitter.com/engineering/en_us/topics/insights/2016/discovery-and-consumption-of-analytics-data-at-twitter.html

https://blog.twitter.com/engineering/en_us/topics/infrastructure/2019/interactive-analytics-at-mopub.html

http://www.vldb.org/pvldb/vol6/p1138-elmeleegy.pdf

http://www.vldb.org/pvldb/vol9/p1269-liu.pdf

https://www.vldb.org/pvldb/vldb2010/pvldb_vol3/I08.pdf

https://research.fb.com/publications/realtime-data-processing-at-facebook/

https://research.fb.com/publications/presto-sql-on-everything/

https://research.fb.com/publications/a-large-scale-study-of-data-center-network-reliability/

https://research.fb.com/publications/providing-streaming-joins-as-a-service-at-facebook/

https://research.fb.com/publications/sve-distributed-video-processing-at-facebook-scale/

https://research.fb.com/publications/canopy-end-to-end-performance-tracing-at-scale/

References (6 of 12)

https://research.fb.com/publications/dqbarge-improving-data-quality-tradeoffs-in-large-scale-internet-services-2/

https://research.fb.com/publications/cubrick-indexing-millions-of-records-per-second-for-interactive-analytics/

https://research.fb.com/publications/one-trillion-edges-graph-processing-at-facebook-scale/

https://research.fb.com/publications/cubrick-a-scalable-distributed-molap-database-for-fast-analytics/

https://research.fb.com/publications/wormhole-reliable-pub-sub-to-support-geo-replicated-internet-services/

https://research.fb.com/publications/an-analysis-of-facebook-photo-caching/

https://research.fb.com/publications/unicorn-a-system-for-searching-the-social-graph/

https://research.fb.com/publications/tao-facebooks-distributed-data-store-for-the-social-graph-2/

https://research.fb.com/publications/tao-facebooks-distributed-data-store-for-the-social-graph/

https://research.fb.com/publications/apache-hadoop-goes-realtime-at-facebook/

https://research.fb.com/publications/finding-a-needle-in-haystack-facebooks-photo-storage/

https://research.fb.com/publications/data-warehousing-and-analytics-infrastructure-at-facebook/

https://research.fb.com/publications/hive-a-warehousing-solution-over-a-map-reduce-framework/

https://research.fb.com/publications/applied-machine-learning-at-facebook-a-datacenter-infrastructure-perspective/

References (7 of 12)

https://research.fb.com/publications/facebooks-data-center-network-architecture/

https://research.fb.com/publications/presto-sql-on-everything/

https://research.fb.com/wp-content/uploads/2016/11/realtime_data_processing_at_facebook.pdf

https://engineering.linkedin.com/distributed-systems/log-what-every-software-engineer-should-know-about-real-time-datas-unifying

https://engineering.linkedin.com/blog/topic/stream-processing

https://engineering.linkedin.com/architecture/brief-history-scaling-linkedin

https://engineering.linkedin.com/blog/2016/06/voices--a-text-analytics-platform-for-understanding-member-feedb

https://engineering.linkedin.com/blog/topic/espresso

https://engineering.linkedin.com/teams/data/projects/data-management

https://engineering.linkedin.com/teams/data/projects/wherehows

https://engineering.linkedin.com/blog/2016/03/open-sourcing-wherehows--a-data-discovery-and-lineage-portal

https://engineering.linkedin.com/teams/data/projects/pinot

https://engineering.linkedin.com/teams/data/projects/xlnt

https://engineering.linkedin.com/ab-testing/xlnt-platform-driving-ab-testing-linkedin

References (8 of 12)

https://engineering.linkedin.com/teams/data/projects/dr-elephant

https://engineering.linkedin.com/teams/data/projects/gobblin

https://engineering.linkedin.com/teams/data/projects/dali

https://engineering.linkedin.com/blog/topic/Spark

https://engineering.linkedin.com/blog/topic/hadoop

https://docs.microsoft.com/en-us/azure/architecture/data-guide/

https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/

https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/batch-processing

https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/real-time-processing

https://docs.microsoft.com/en-us/azure/architecture/data-guide/technology-choices/analytical-data-stores

https://azure.microsoft.com/en-us/solutions/architecture/advanced-analytics-on-big-data/

https://azure.microsoft.com/en-us/solutions/architecture/modern-data-warehouse/

https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/

https://blogs.msdn.microsoft.com/azurecat/2018/12/24/azure-data-architecture-guide-blog-8-data-warehousing/

References (9 of 12)

https://docs.microsoft.com/en-us/azure/architecture/guide/architecture-styles/big-data

http://www.vldb.org/pvldb/vol12/p2170-tan.pdf

https://www.vldb.org/pvldb/vldb2010/pvldb_vol3/I07.pdf

http://www.vldb.org/pvldb/vol12/p2143-antonopoulos.pdf

https://azure.microsoft.com/en-us/blog/microsoft-azure-data-welcomes-attendees-to-vldb-2018/

http://www.vldb.org/pvldb/vol9/p1245-lang.pdf

http://www.vldb.org/pvldb/vol8/p401-chandramouli.pdf

http://www.vldb.org/pvldb/vol6/p1178-lomet.pdf

http://www.cs.ucf.edu/~kienhua/classes/COP5711/Papers/MSazure2017.pdf

https://www.microsoft.com/en-us/research/uploads/prod/2019/05/socrates.pdf

https://www.microsoft.com/en-us/research/uploads/prod/2019/05/QuickInsights-camera-ready-final.pdf

http://pages.cs.wisc.edu/~remzi/Classes/739/Spring2004/Papers/p215-dageville-snowflake.pdf

http://www.vldb.org/pvldb/vol11/p1317-mahajan.pdf

http://www.vldb.org/pvldb/vol12/p681-zou.pdf

References (10 of 12)

http://www.benstopford.com/2012/10/28/the-best-of-vldb-2012/

http://pages.cs.wisc.edu/~jignesh/publ/Quickstep.pdf

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.640.7744&rep=rep1&type=pdf

https://www.betterevaluation.org/sites/default/files/data_cleaning.pdf

http://www.vldb.org/pvldb/vol6/p1009-aji.pdf

https://www.scitepress.org/papers/2017/65871/65871.pdf

https://db.in.tum.de/~kipf/papers/fastdata.pdf

https://people.csail.mit.edu/matei/papers/2015/sigmod_spark_sql.pdf

https://cs.stanford.edu/~matei/papers/2013/sigmod_shark.pdf

https://hal.archives-ouvertes.fr/tel-02059437v2/document

https://arxiv.org/pdf/1907.00050.pdf

http://www.cs.cornell.edu/~ragarwal/pubs/zipg.pdf

http://dbgroup.cs.tsinghua.edu.cn/ligl/papers/sigmod19-tutorial-paper.pdf

https://homes.cs.washington.edu/~magda/papers/wang-cidr17.pdf

References (11 of 12)

https://arxiv.org/pdf/1702.01596.pdf

https://www.comp.nus.edu.sg/~memepic/pdfs/mem-survey.pdf

http://www.vldb.org/pvldb/vol11/p1414-agrawal.pdf

http://da.qcri.org/rheem/pdf/paper/rheemtutorial.pdf

http://www.ijsrp.org/research-paper-0613/ijsrp-p18143.pdf

https://www.slideshare.net/YunyaoLi/enterprise-search-in-the-big-data-era-recent-developments-and-open-challenges-38801004

http://cidrdb.org/cidr2017/papers/p123-barber-cidr17.pdf

https://adalabucsd.github.io/papers/Slides_2017_Tutorial_SIGMOD.pdf

http://static.druid.io/docs/druid.pdf

http://www.vldb.org/pvldb/vol11/p1822-cai.pdf

http://cs.brown.edu/~xhu3/paper/diffprivacy.pdf

https://e.huawei.com/us/solutions/cloud-computing/big-data

http://www.vldb.org/pvldb/vol7/p1754-huawei.pdf

https://dl.acm.org/citation.cfm?id=3275550

References (12 of 12)

http://conferences.cis.umac.mo/icde2019/wp-content/uploads/2019/06/icde-2019-keynote-jianjun-chen.pdf

https://arxiv.org/pdf/1907.00050.pdf

https://astro.temple.edu/~tua86772/zhang14SIGMOD_DEMO.pdf

https://users.wpi.edu/~yli15/Includes/SIGMOD15Telco.pdf

https://15721.courses.cs.cmu.edu/spring2016/papers/p337-soliman.pdf

https://aws.amazon.com/big-data/datalakes-and-analytics/

http://www.vldb.org/pvldb/vol11/p1781-schelter.pdf

https://www.allthingsdistributed.com/files/p1041-verbitski.pdf

https://dl.acm.org/citation.cfm?id=3196937

https://lovvge.github.io/Forecasting-Tutorial-VLDB-2018/

http://lunadong.com/

https://databricks.com/wp-content/uploads/2018/12/adam-sigmod-2015.pdf

End of tutorial -- Thank you