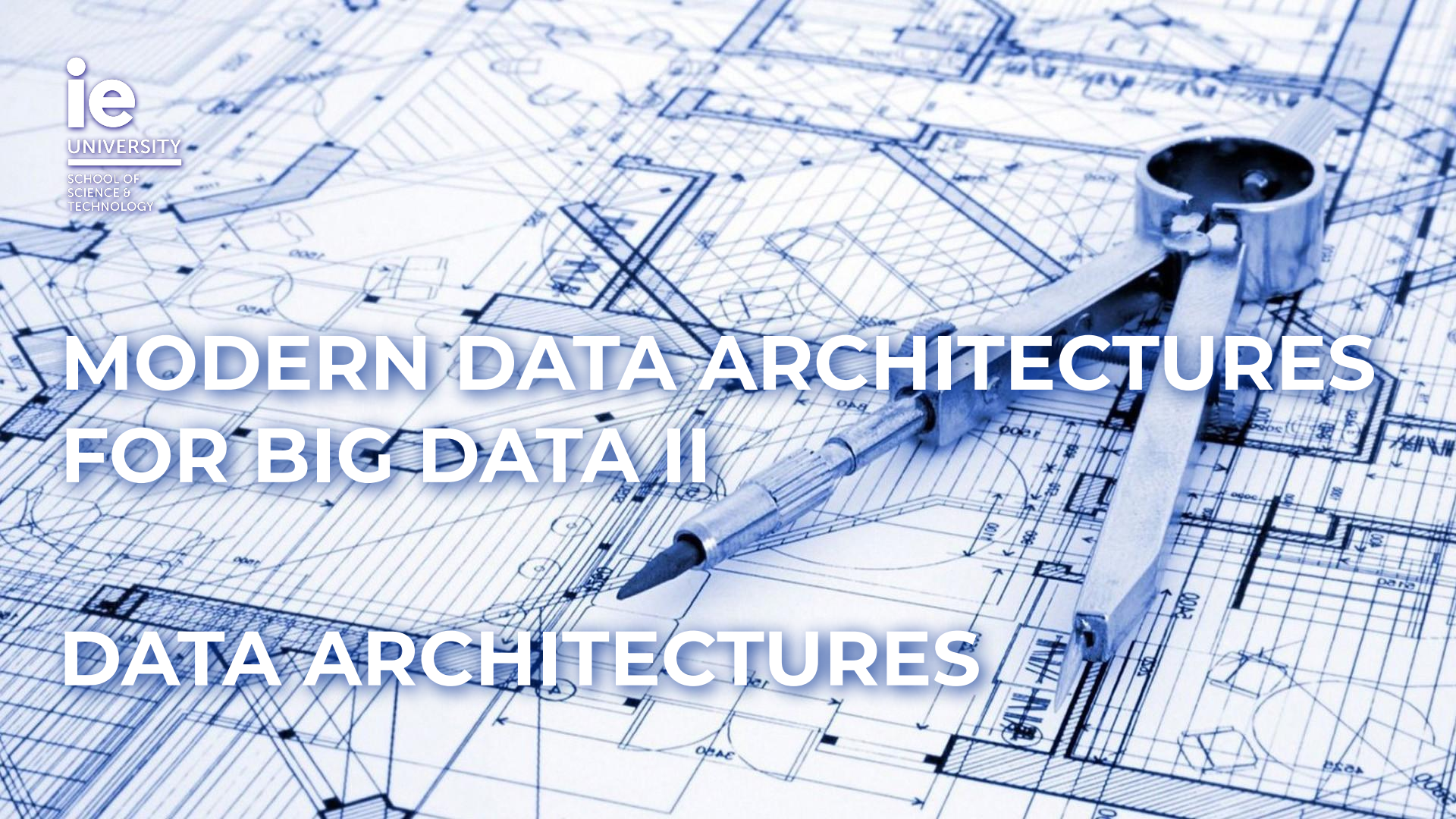
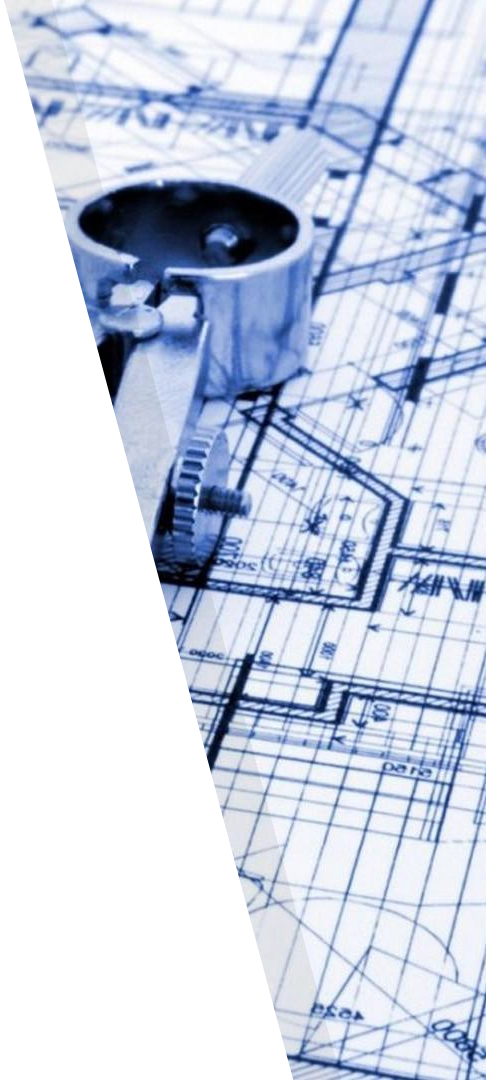

ie
UNIVERSITY
SCHOOL OF
SCIENCE &
TECHNOLOGY
MODERN DATA ARCHITECTURES
FOR BIG DATA II
DATA ARCHITECTURES

Agenda

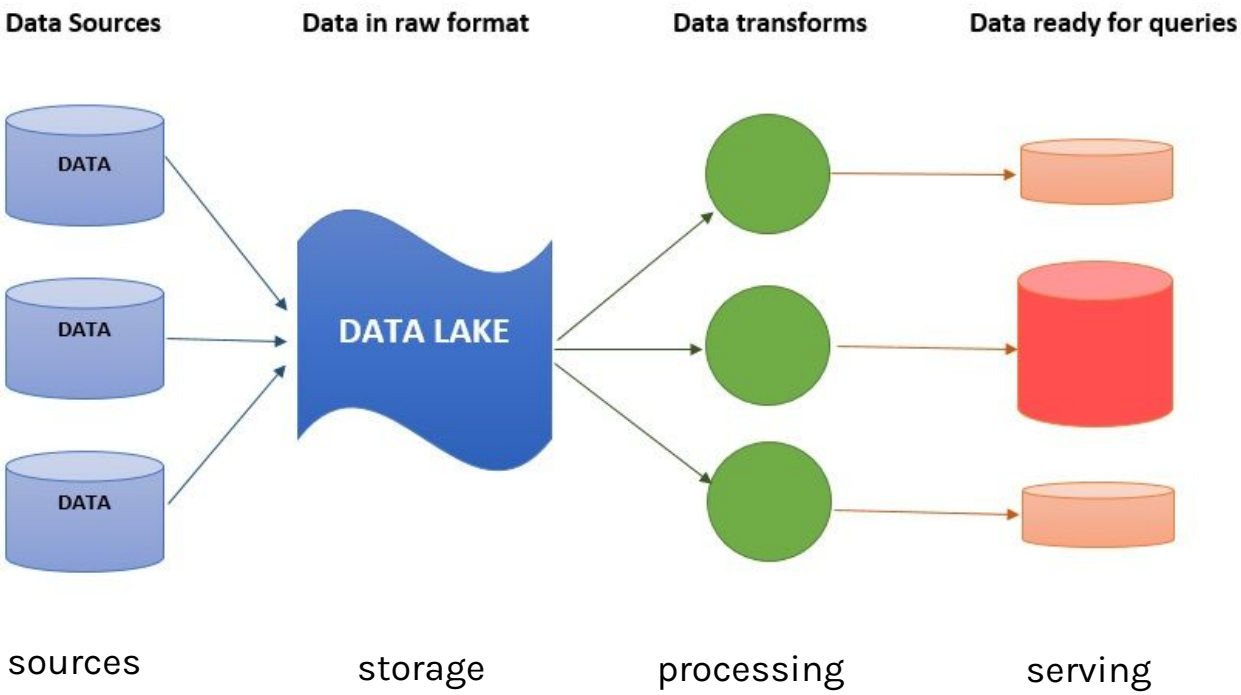
- Data Architecture Patterns
  - Lambda Architecture
  - Kappa Architecture
  - Delta Architecture (LakeHouse)
  - Data Mesh
  - Examples
- Enterprise Data Architectures
  - On premise
  - On cloud
  - Hybrid
- Data Position & Roles
- Summary

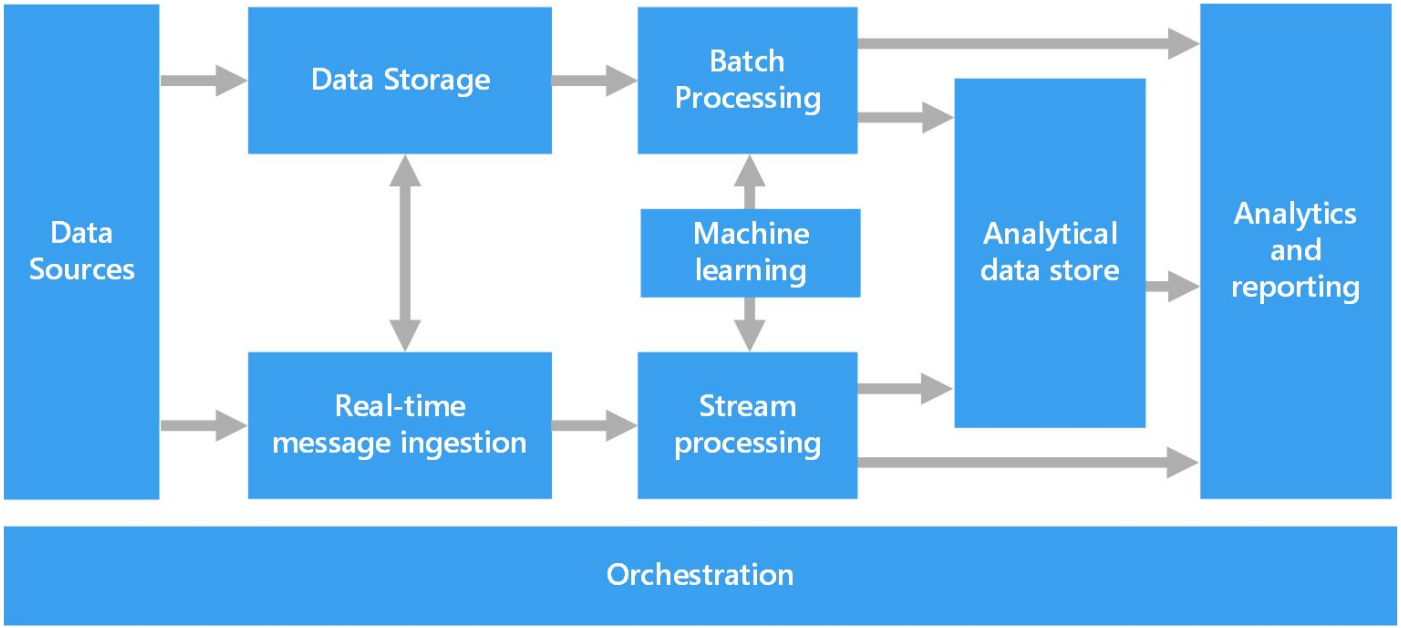


1.

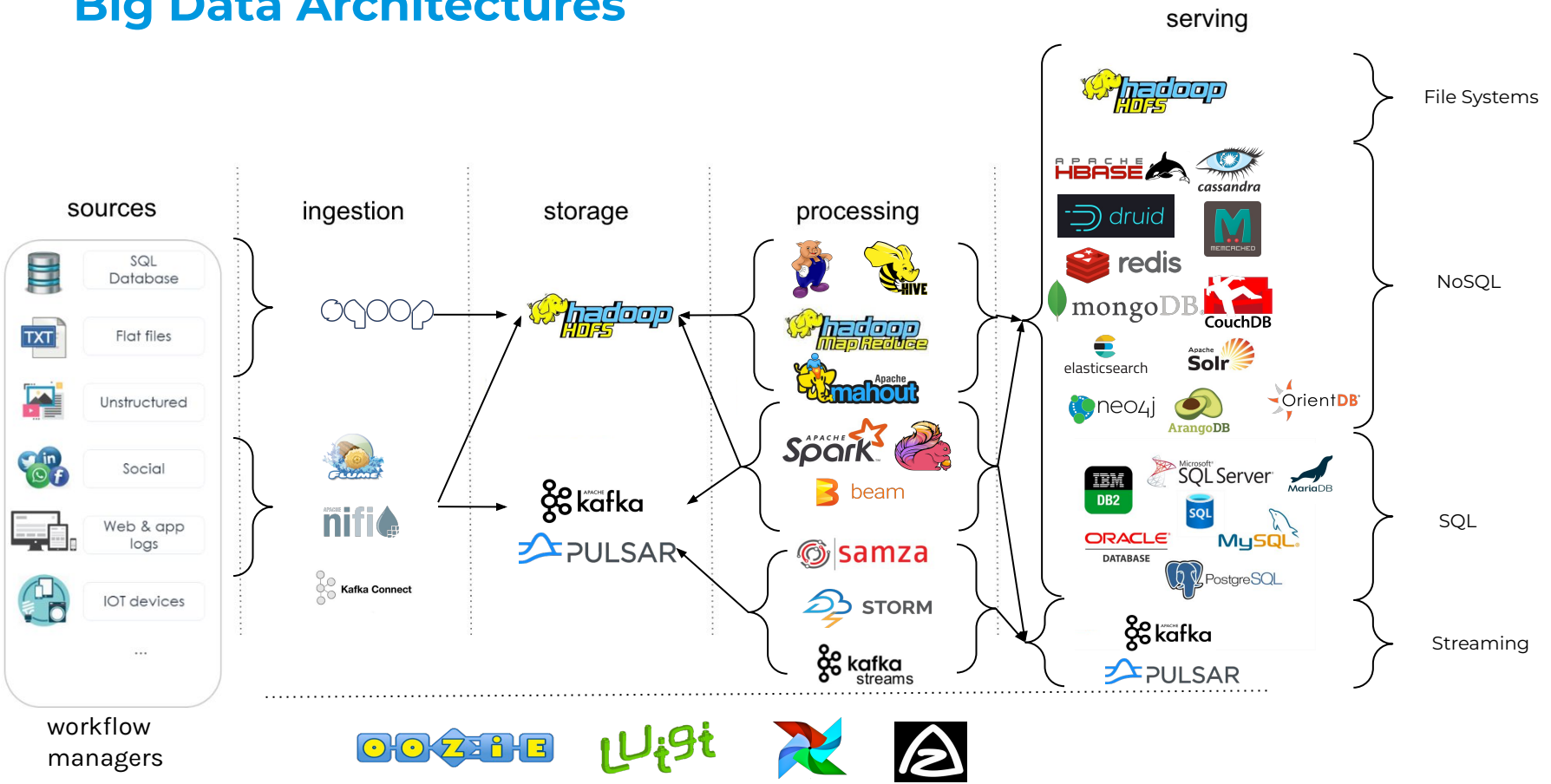
DATA ARCHITECTURE PATTERNS

Big Data Architectures



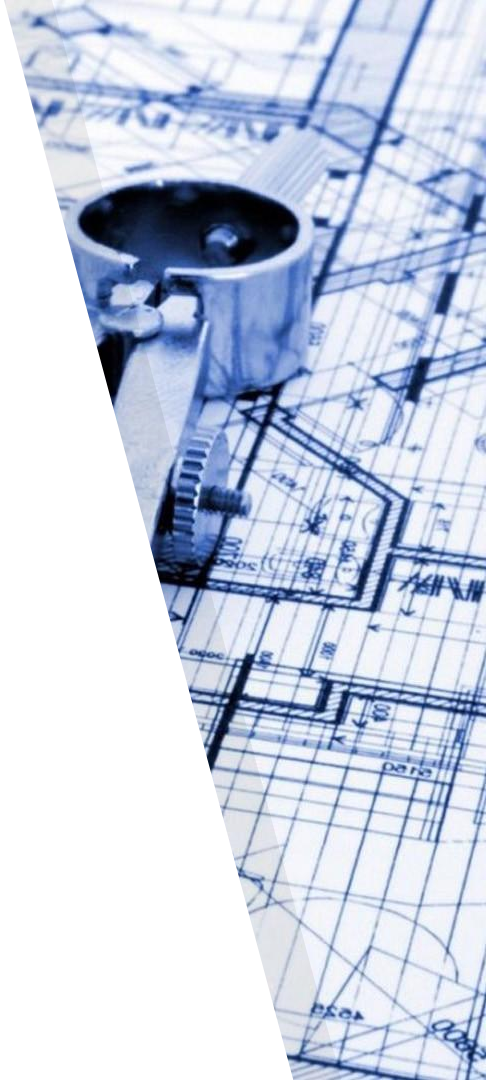

Data Sources
Data Storage
Real-time message ingestion
Batch Processing
Machine learning
Stream processing
Analytical data store
Analytics and reporting
Orchestration

Big Data Architectures



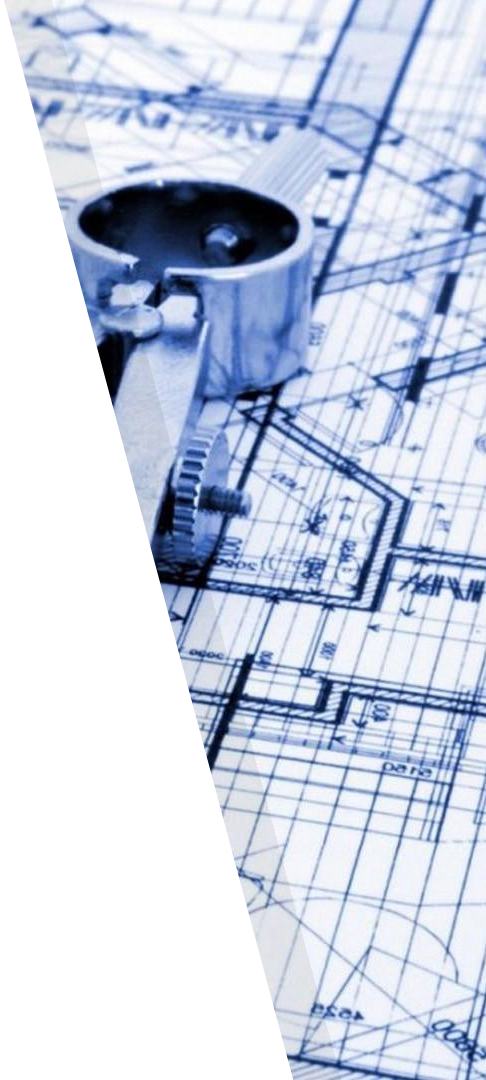
Big Data Architectures

In every big data architecture there is a need for integrating **different data pipelines** with very **different requirements**.



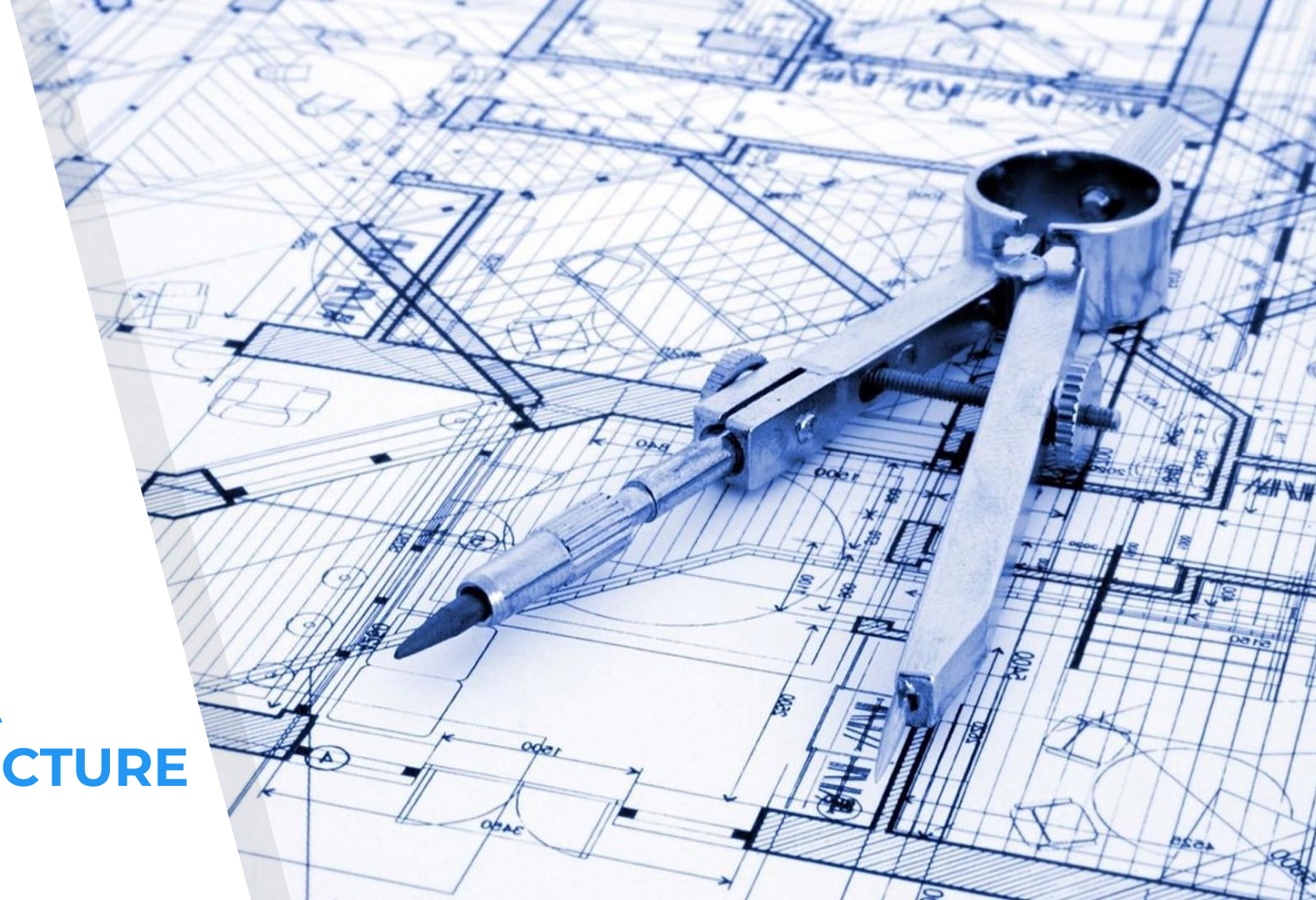
Big Data Architectures

Data architectures are patterns that define how to connect several technologies **to solve different analytical business needs**



1.1

LAMBDA ARCHITECTURE

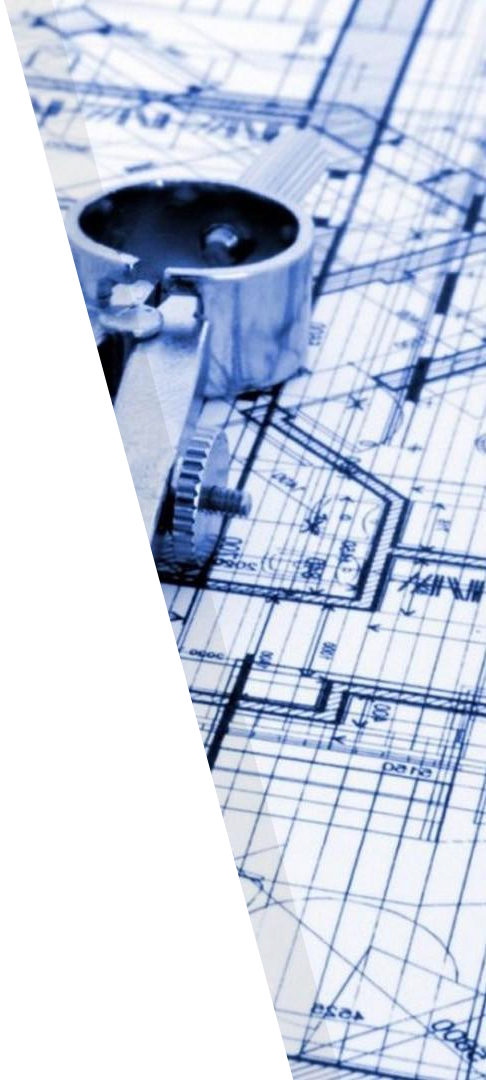


Lambda Architecture

Lambda architecture is an architecture where records are processed by a batch system and streaming system in parallel.

The results are then combined during query/serving time to provide a complete answer.

Separates batch and streaming data flows.



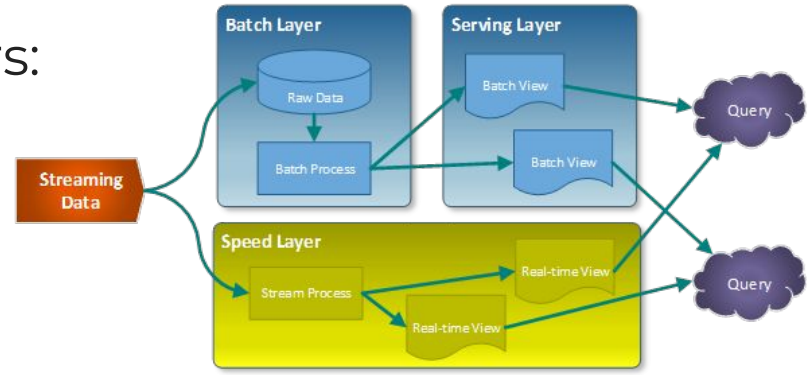
Lambda Architecture

2013/2014

Proposed by Nathan Marz (Twitter)

Comprises three layers:

- batch layer
- speed layer
- serving layer



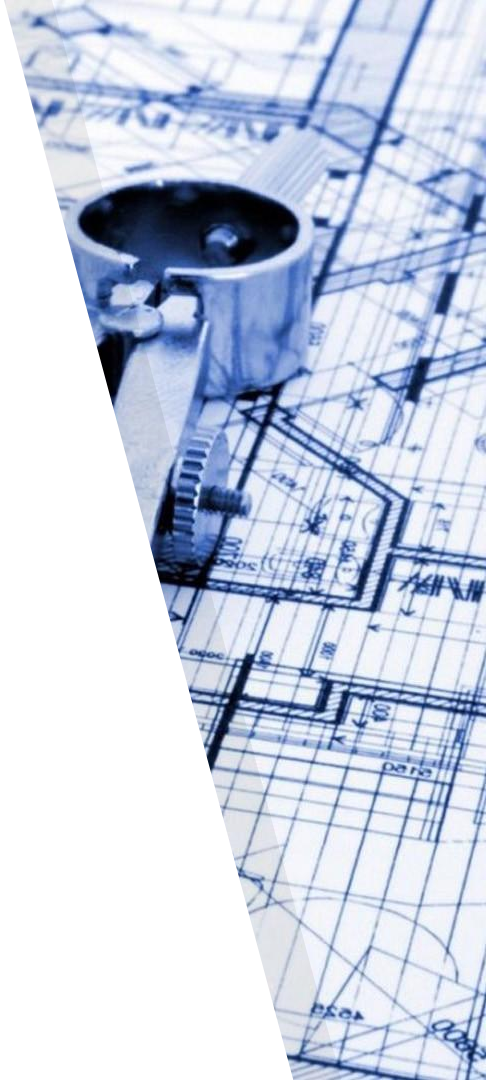
Lambda Architecture Concepts

data:

information that can't be derived from other. (inmutable)

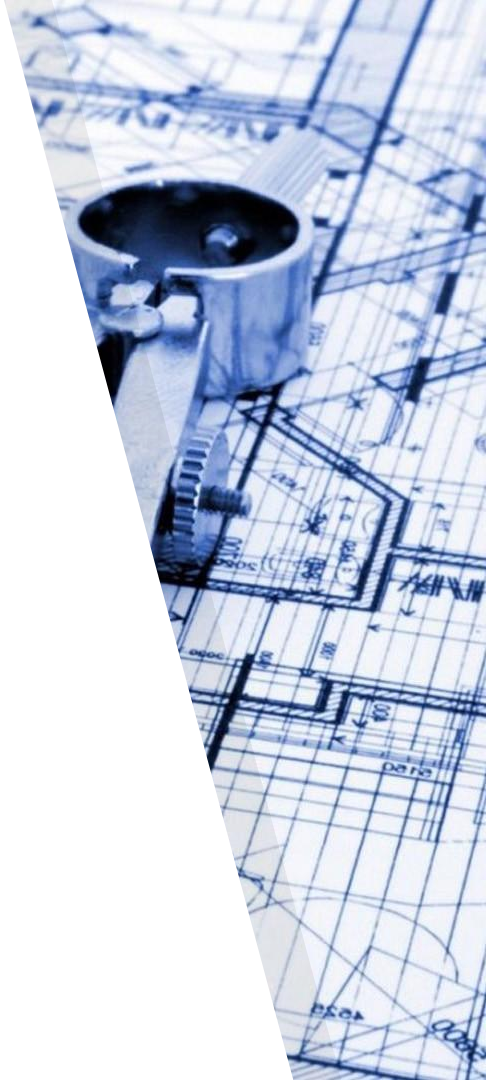
view :

information derived from other base **data** (normally build to answer queries) (can always be regenerated)



Lambda Architecture

All data entering the system **is dispatched** to **both the batch layer** and the **speed layer for processing.**

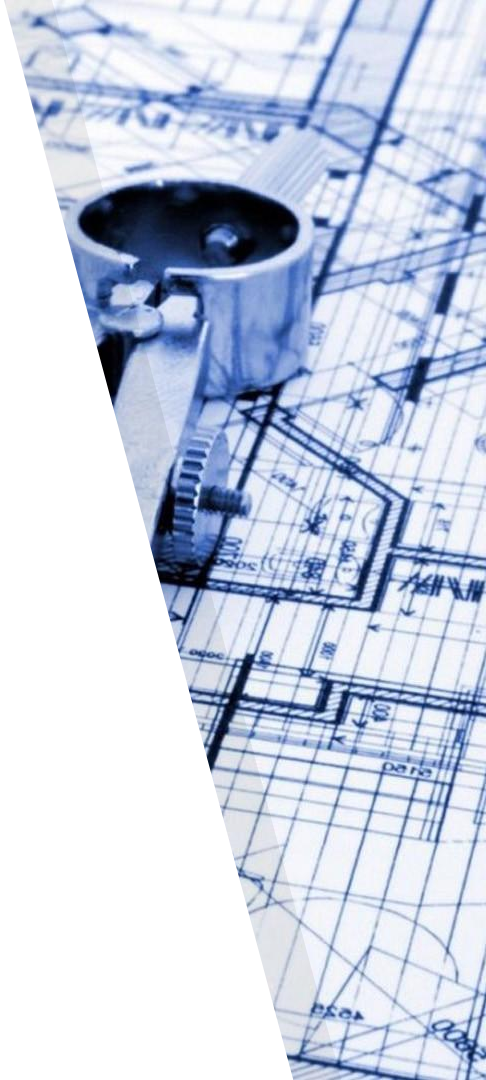


Lambda Architecture Batch Layer

batch layer purpose:

1. **to store** the master copy of datasets (source of truth), an **immutable**, **append-only** set of **raw data**.

2. **to create batch views** from master datasets.



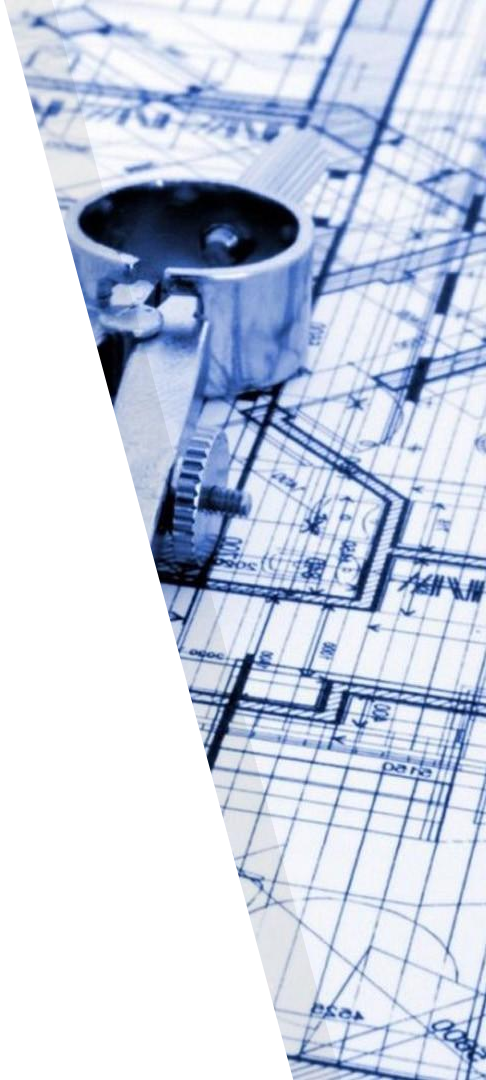
Lambda Architecture Batch Layer

batch layer summary

slow, large and persistent

full recomputation

all data

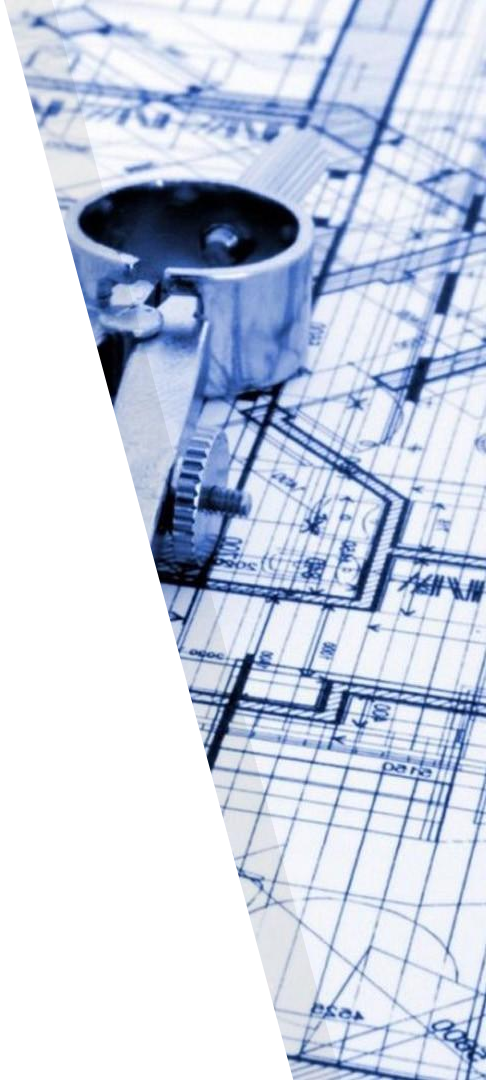


Lambda Architecture Speed Layer

speed layer purpose:

1. to compensate the high latency of updates from batch layer to the serving layer

2. to create **real-time views**



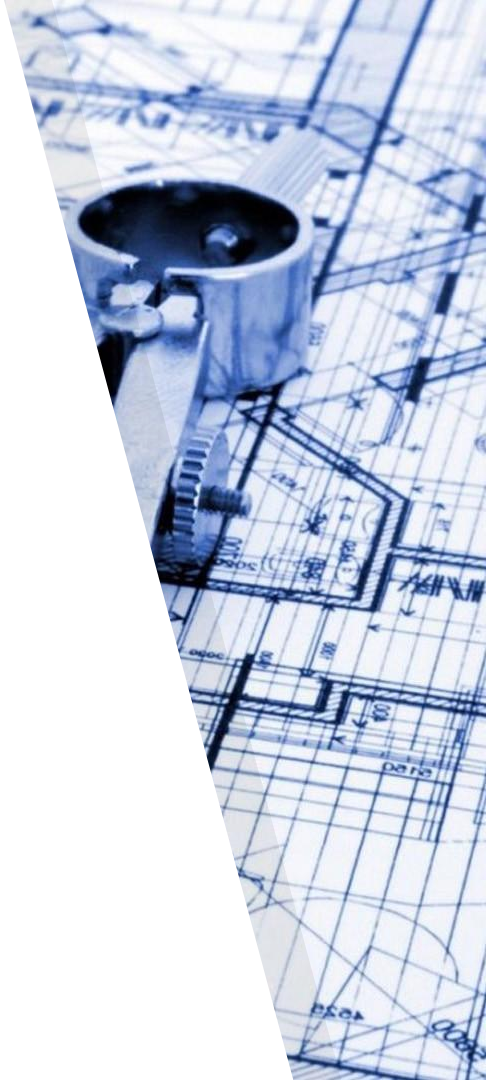
Lambda Architecture Speed Layer

speed layer summary

fast, small and volatile

incremental computation

recent data

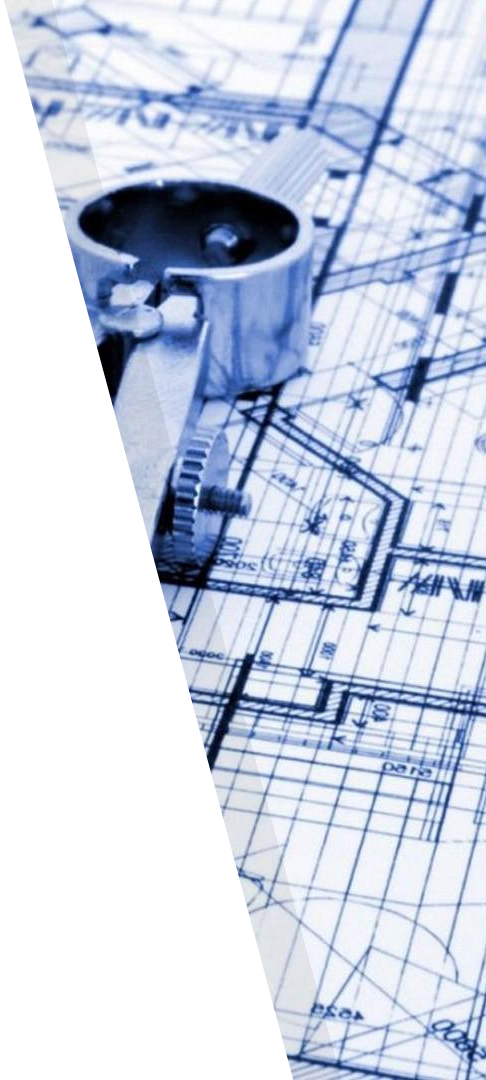


Lambda Architecture Serving Layer

serving layer:

views need to be loaded somewhere in order to be queried.

Often **distributed datastores (NoSQL)** that makes possible random read and batch updates.

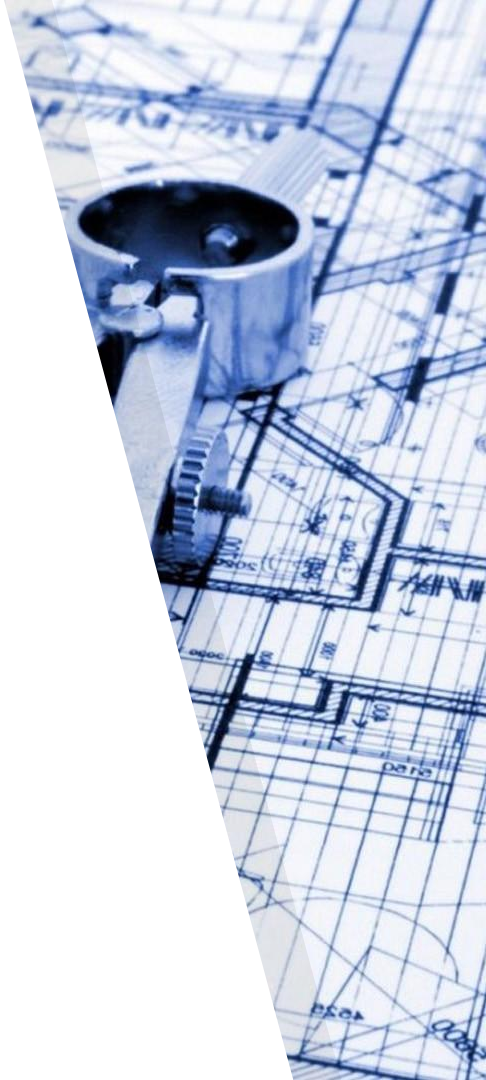


Lambda Architecture Serving Layer

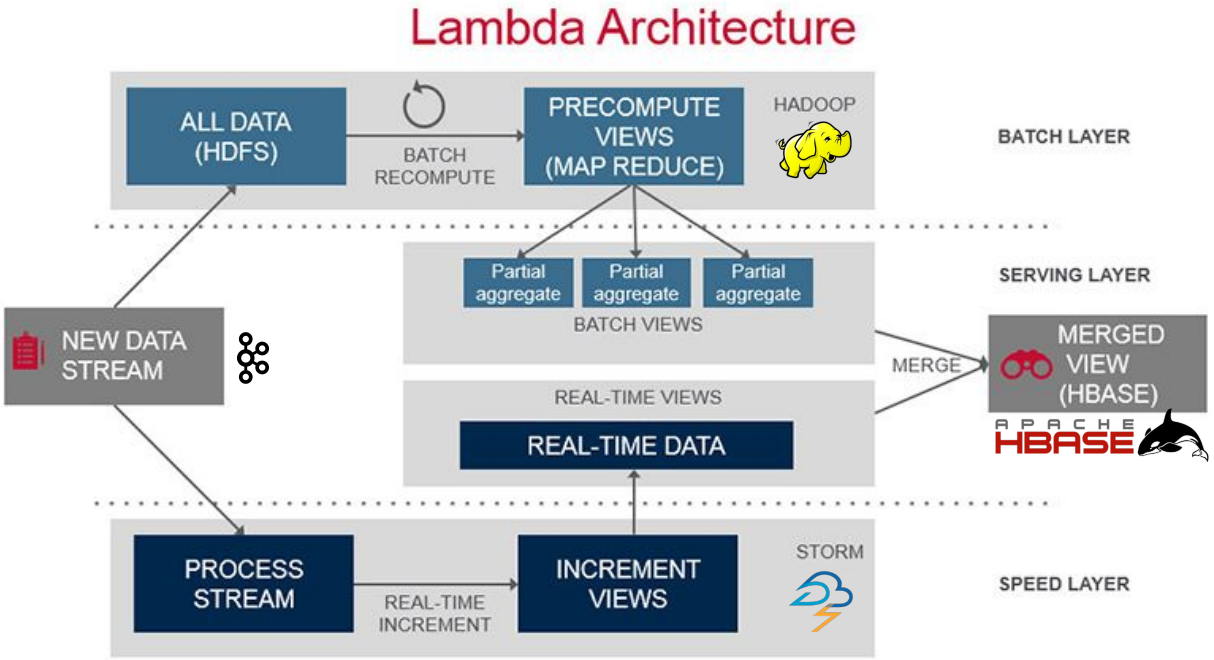
serving layer purpose:

1. **to serve batch views** and **real-time views** so that they can be queried in low-latency way.

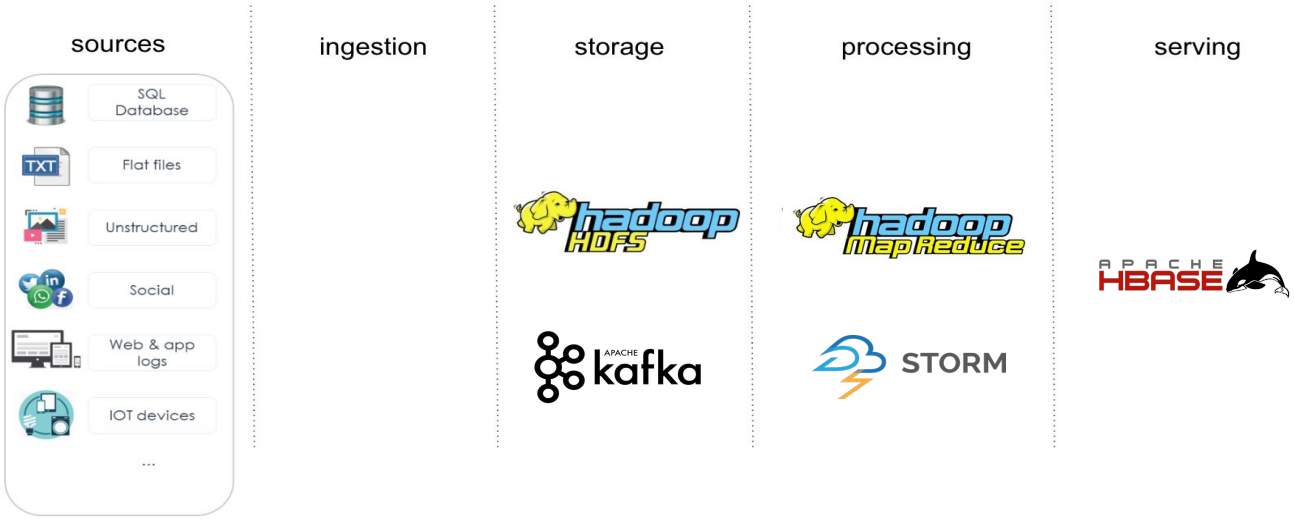
   Any incoming **query** can be answered by **merging results** from **batch views** and **real-time views**.

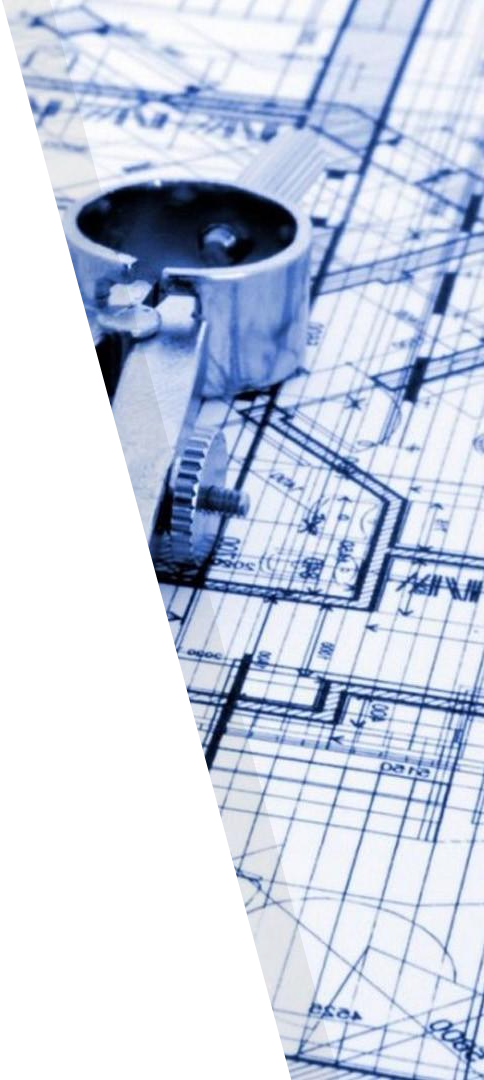


Lambda Architecture



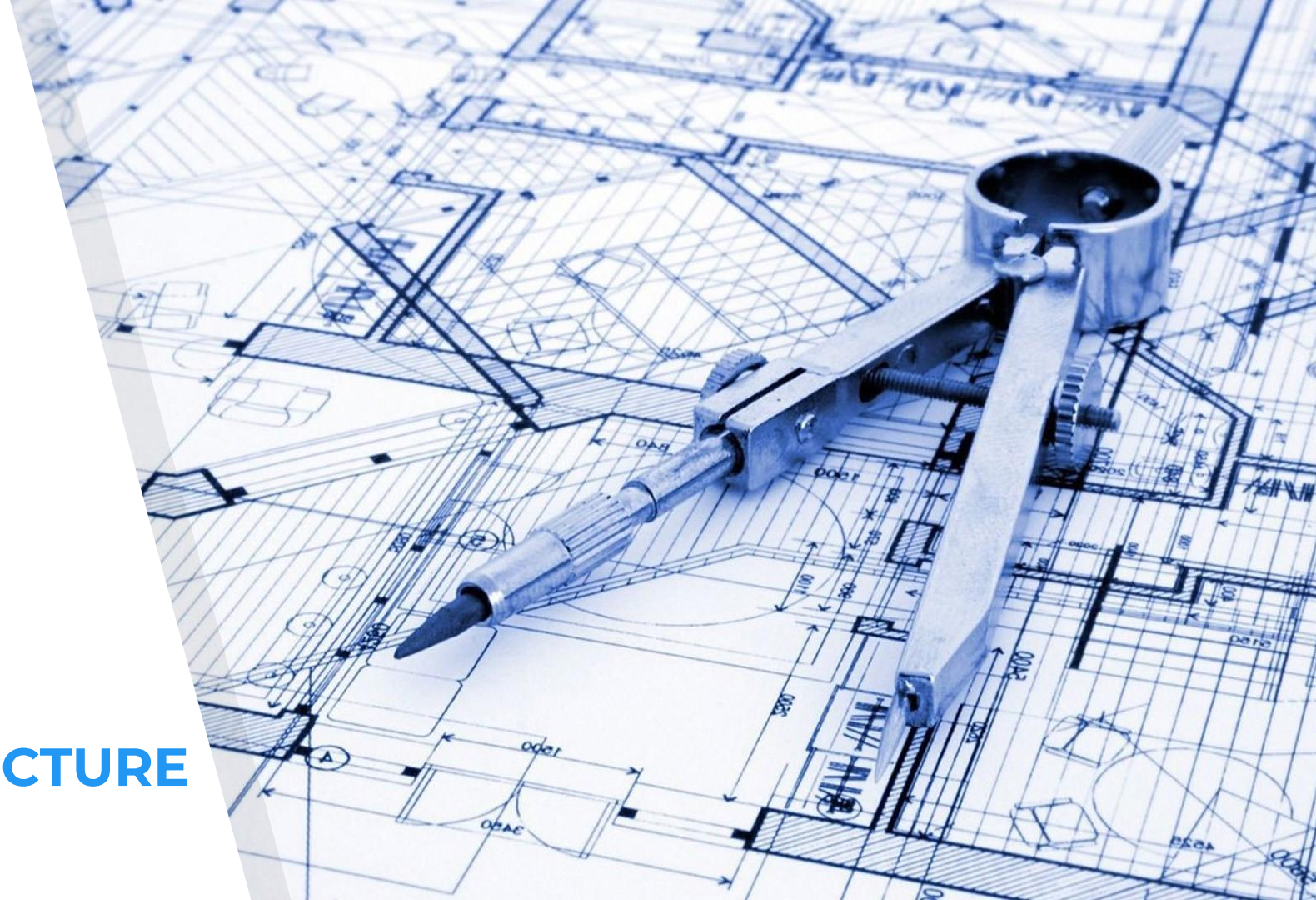
Lambda Architecture





1.2

KAPPA ARCHITECTURE



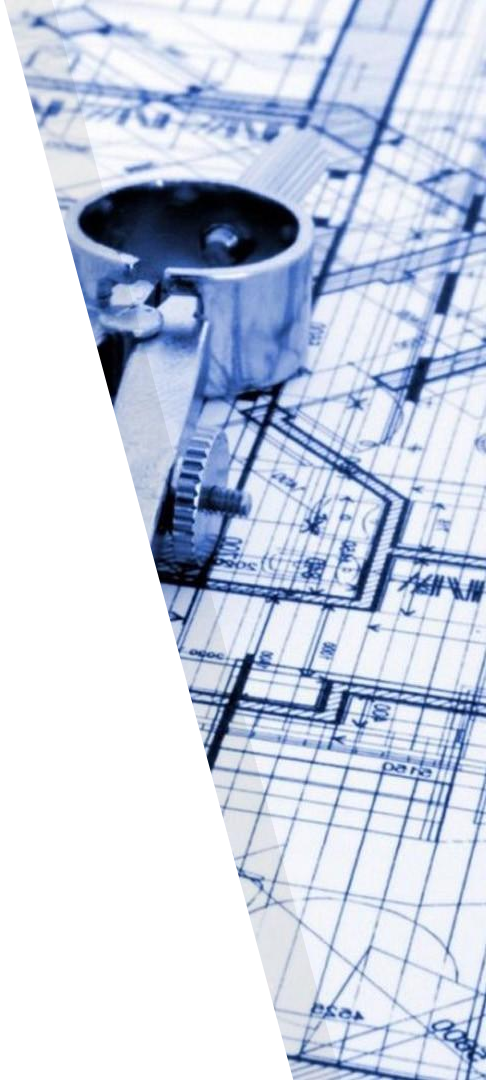
Kappa Architecture

A simplification of Lambda architecture without the batch layer.

Is mainly **focus on real-time data architectures.**

Kappa Architecture cannot be taken as a substitute of Lambda architecture

There are only streaming data flows.



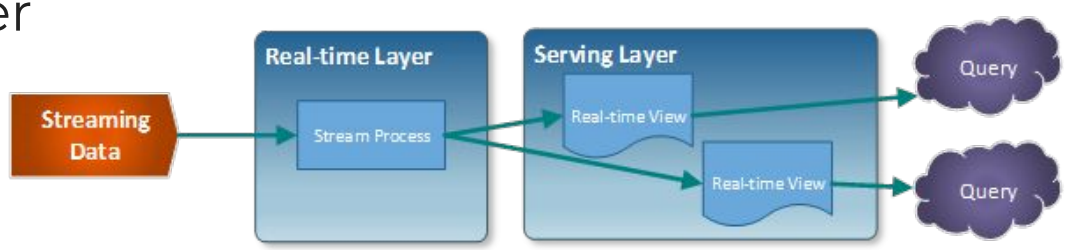
Kappa Architecture

2014/2015

Proposed by Jay Kreps (LinkedIn)

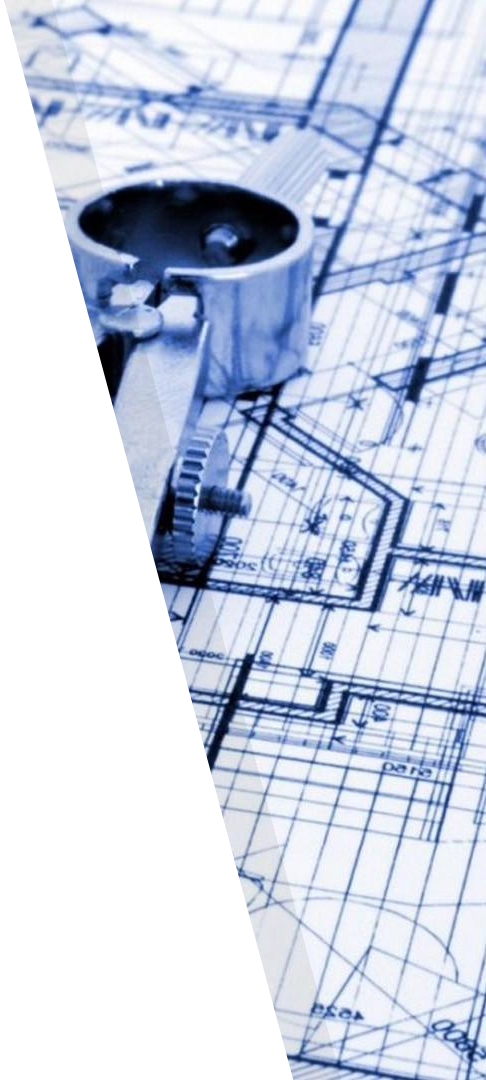
Comprises two layers:

- Real-time layer
- Serving layer



Kappa Architecture Main Ideas

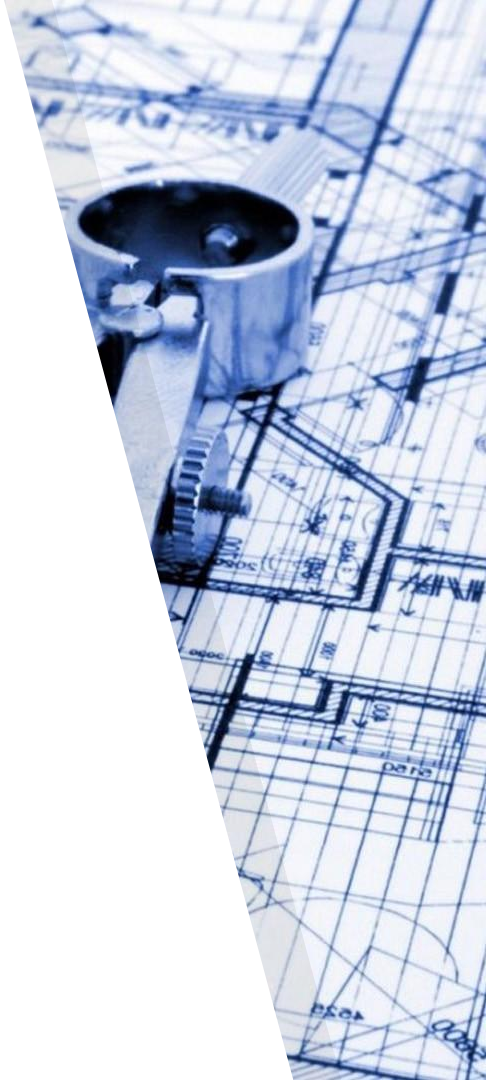
- retain immutable append-only data. (stream storage)
- reprocessing only when code/business logic changes



Kappa Architecture Ideas

All **data** you need **is retained** in a **streaming storage (Kafka).**

For example, if you support to reprocess up to 30 days of data, you set your data retention to 30 days in your Kafka topics.



Kappa Architecture Real-time Layer

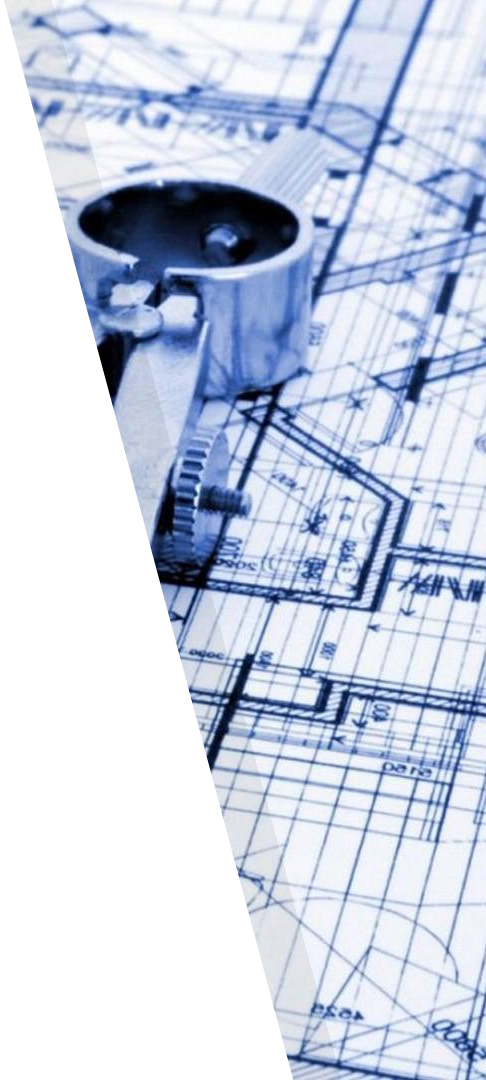
real-time layer:

processes streaming **data** and outputs the results to serving layer

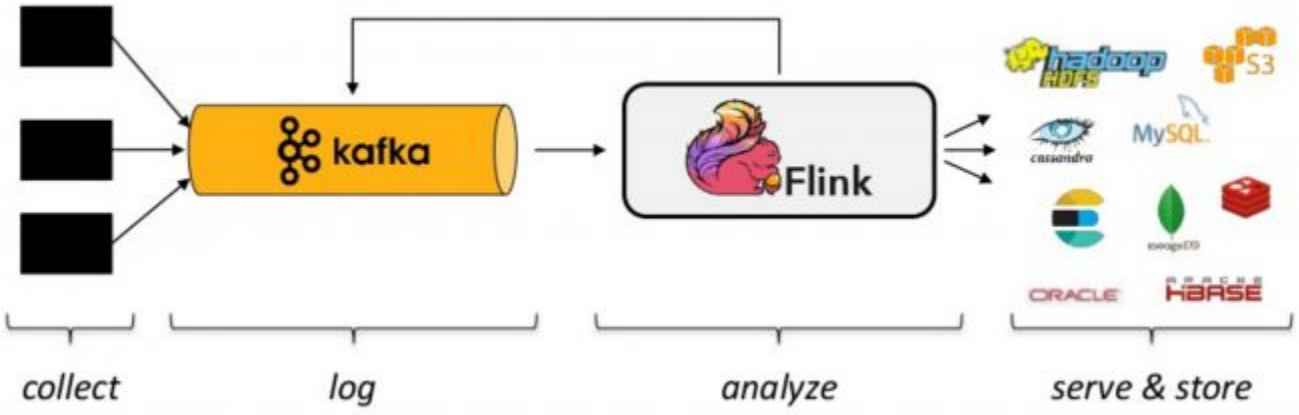
serving layer:

streaming views need to be loaded somewhere in order to be queried.

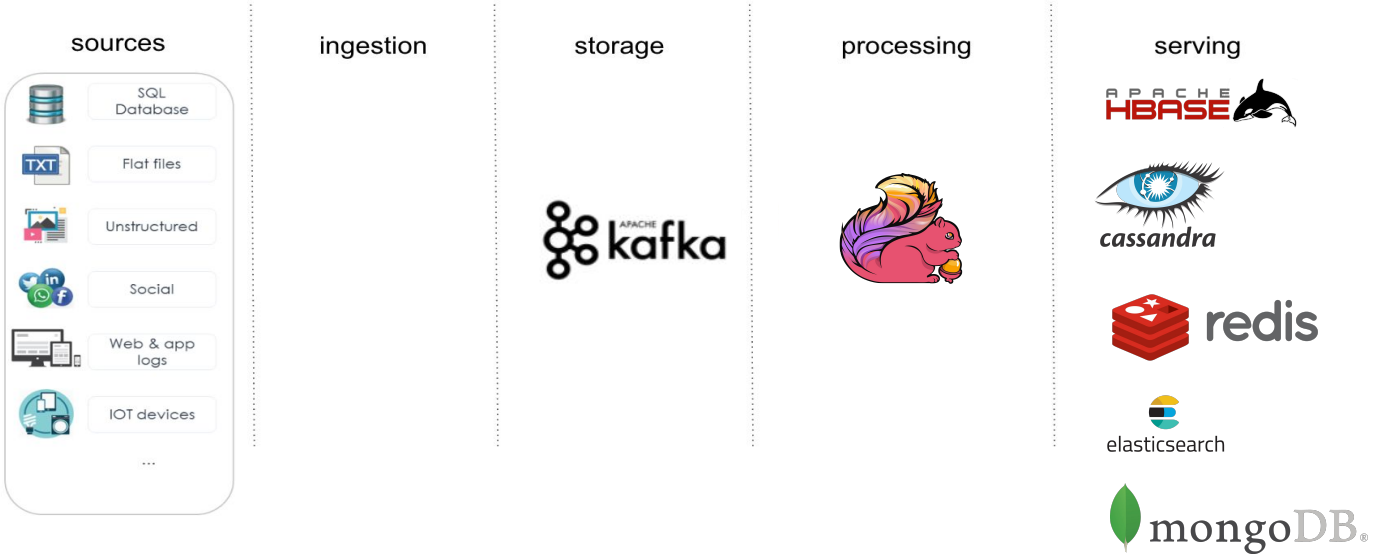
Often **NoSQL datastores** that makes possible random read and batch updates.



Kappa Architecture

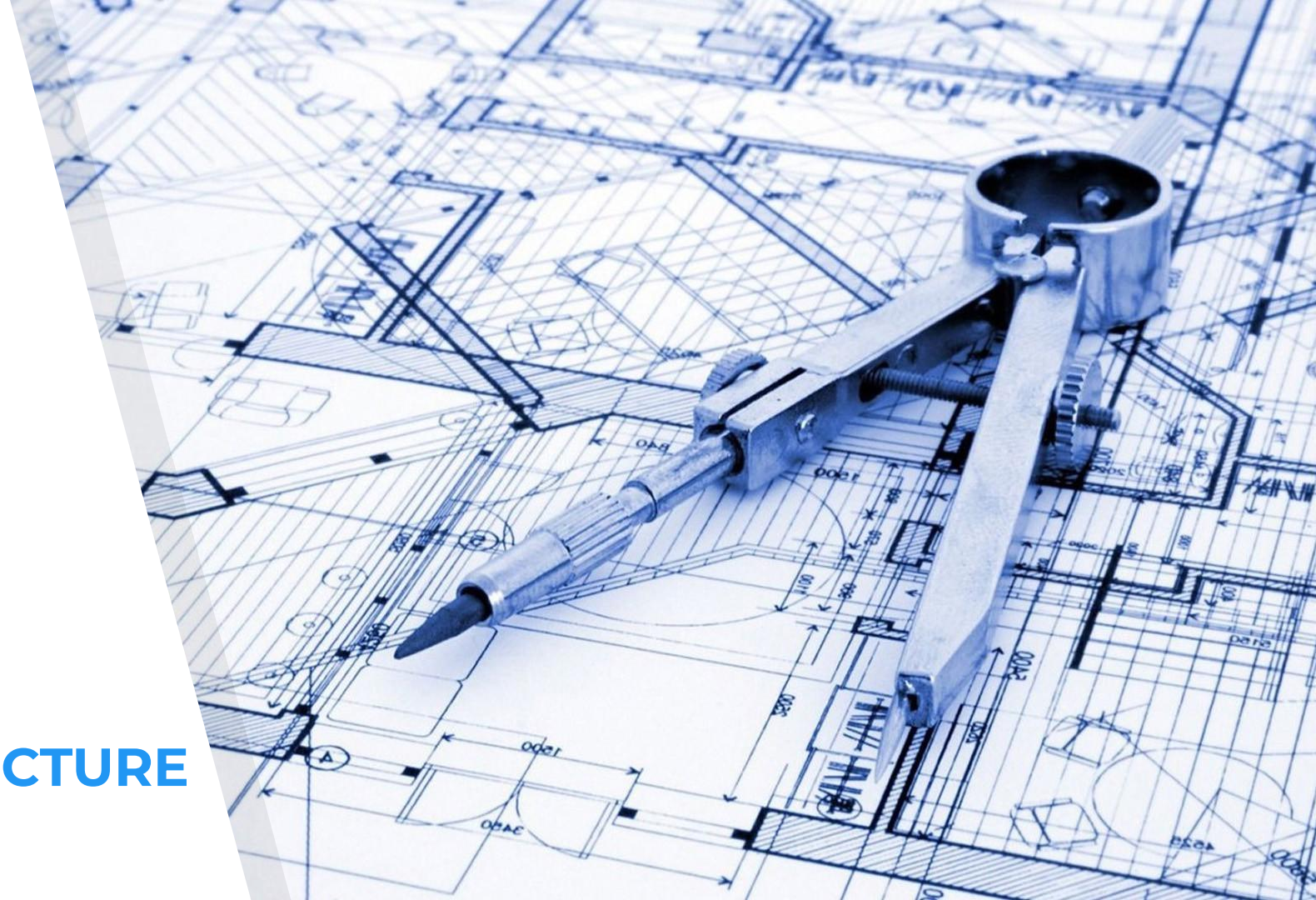


Kappa Architecture



1.3

DELTA ARCHITECTURE



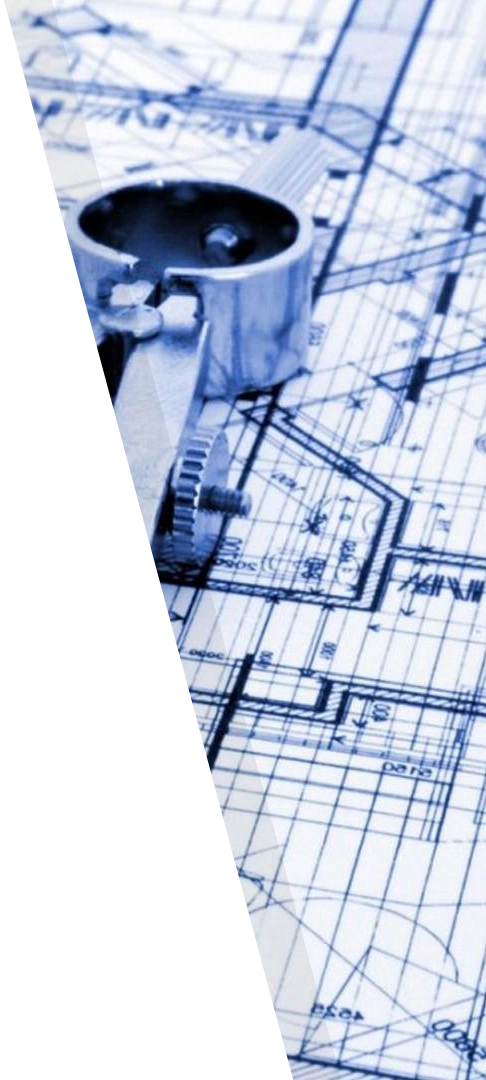
Delta Architecture

2019/2020

Proposed by databricks

Unifies batch and streaming data flows.

Support for incrementally (delta) add new data efficiently in the storage layer that makes possible to process data as soon it arrives.

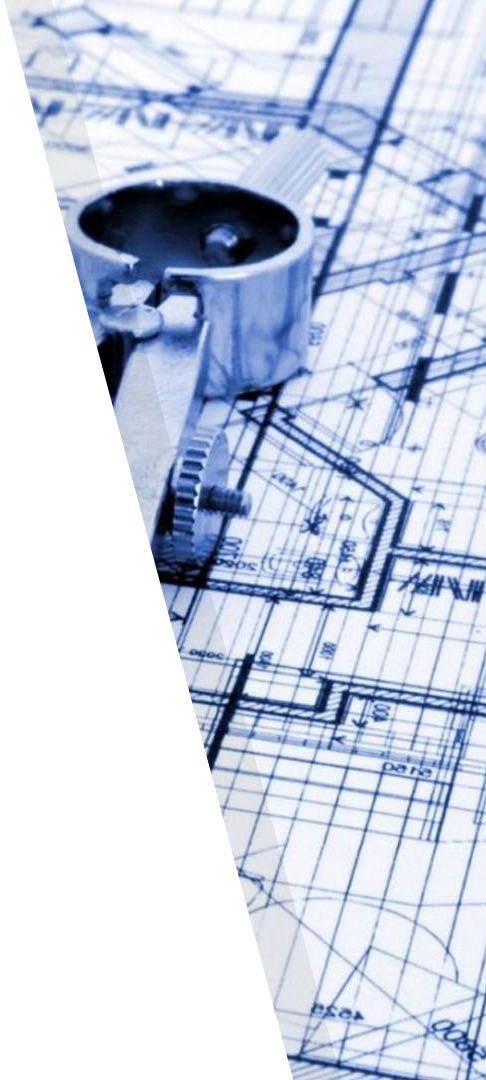


Delta Architecture

2019/2020

Conceptually this architecture patterns is similar to Lambda as it is based on speed and hot path.

The one big difference is that delta architecture no longer considers data lake as immutable

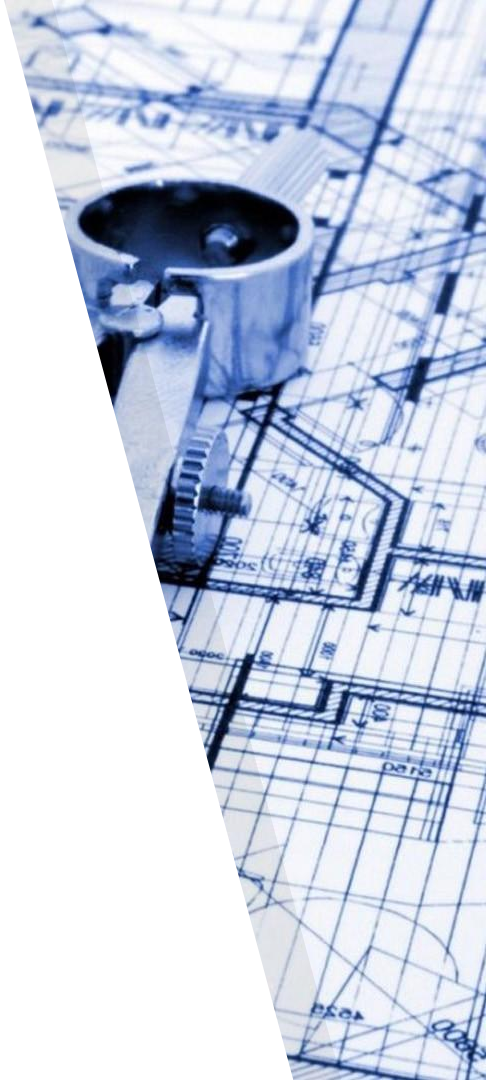


Delta Architecture

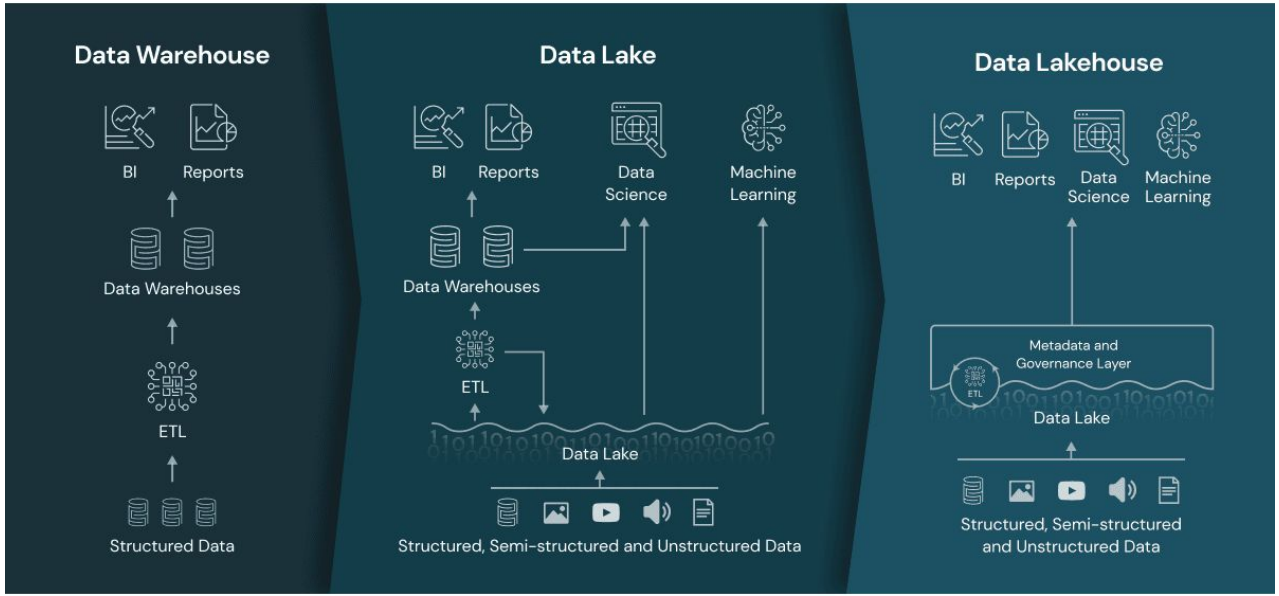
With a new data format called **delta** (based on parquet) is possible to deal with files as they were tables! (updates, inserts, upserts,deletes)

Also makes possible to perform stream processing from tables.

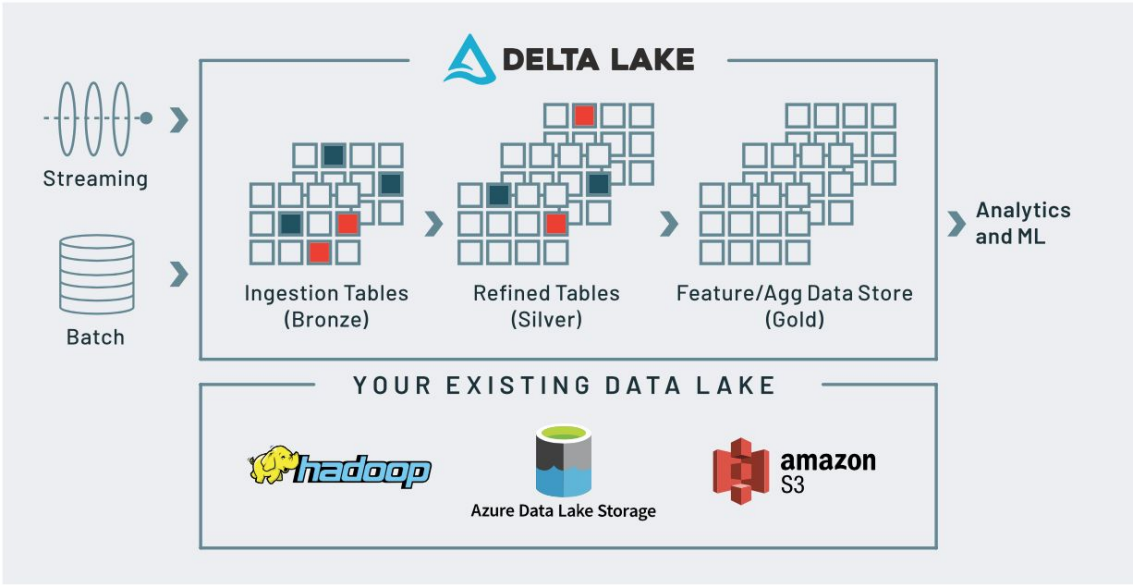
Delta Architecture is also known as **Lakehouse**

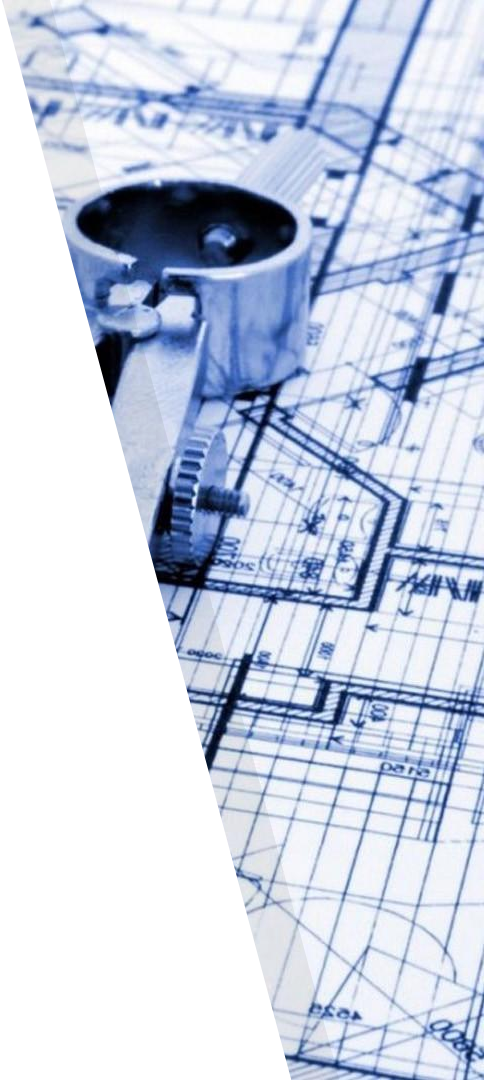


Delta Architecture

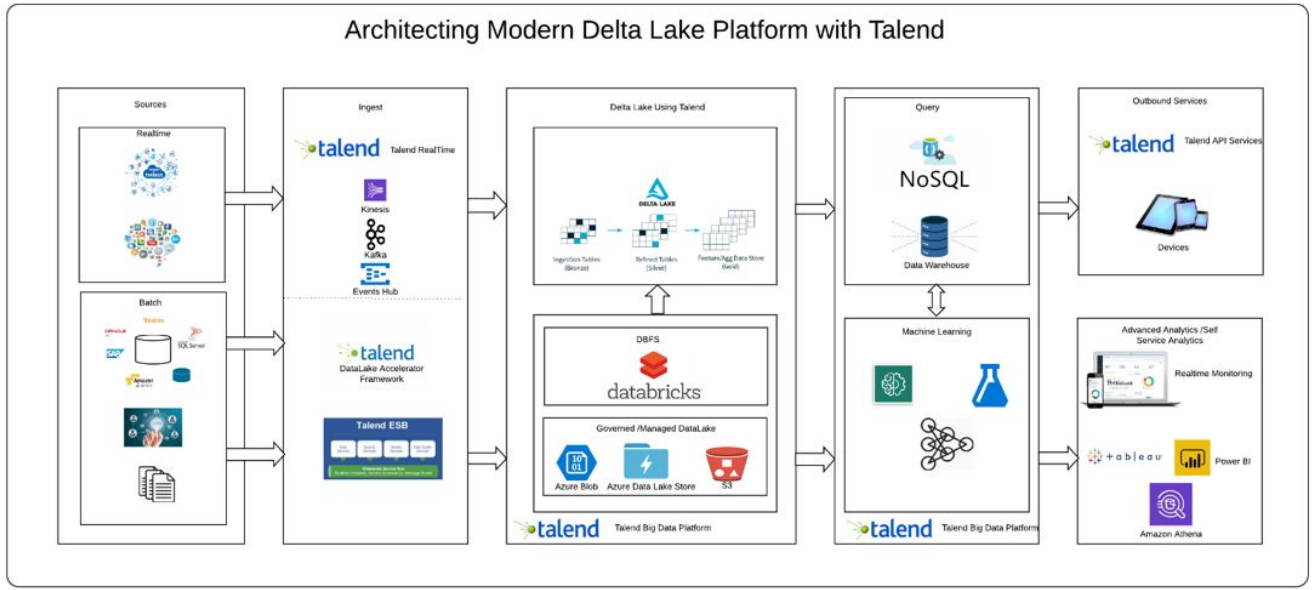


Delta Architecture

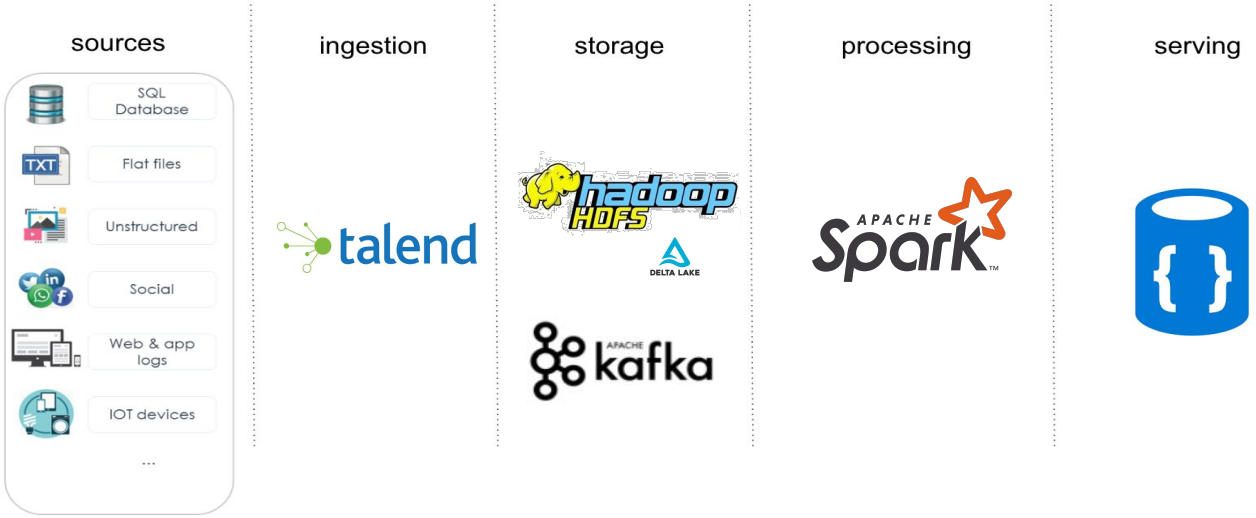


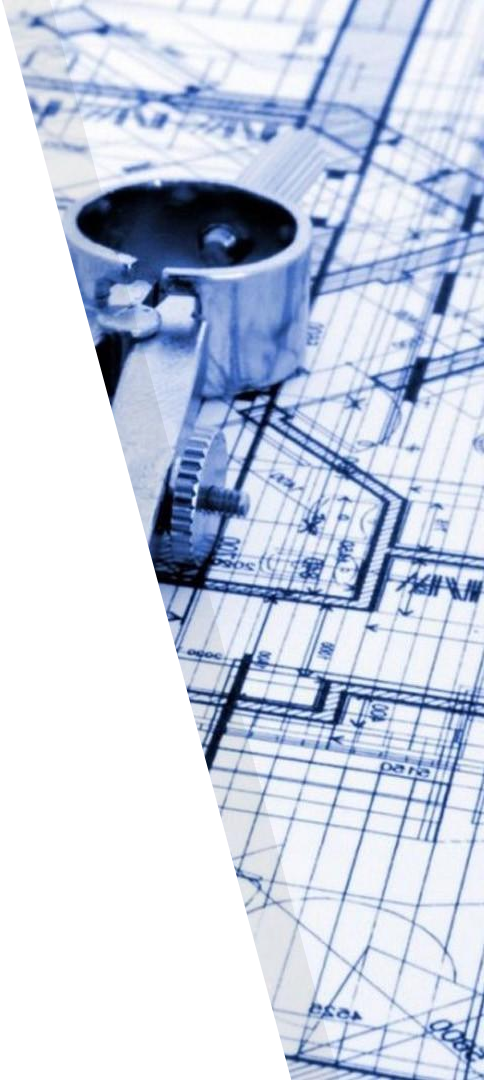


Delta Architecture



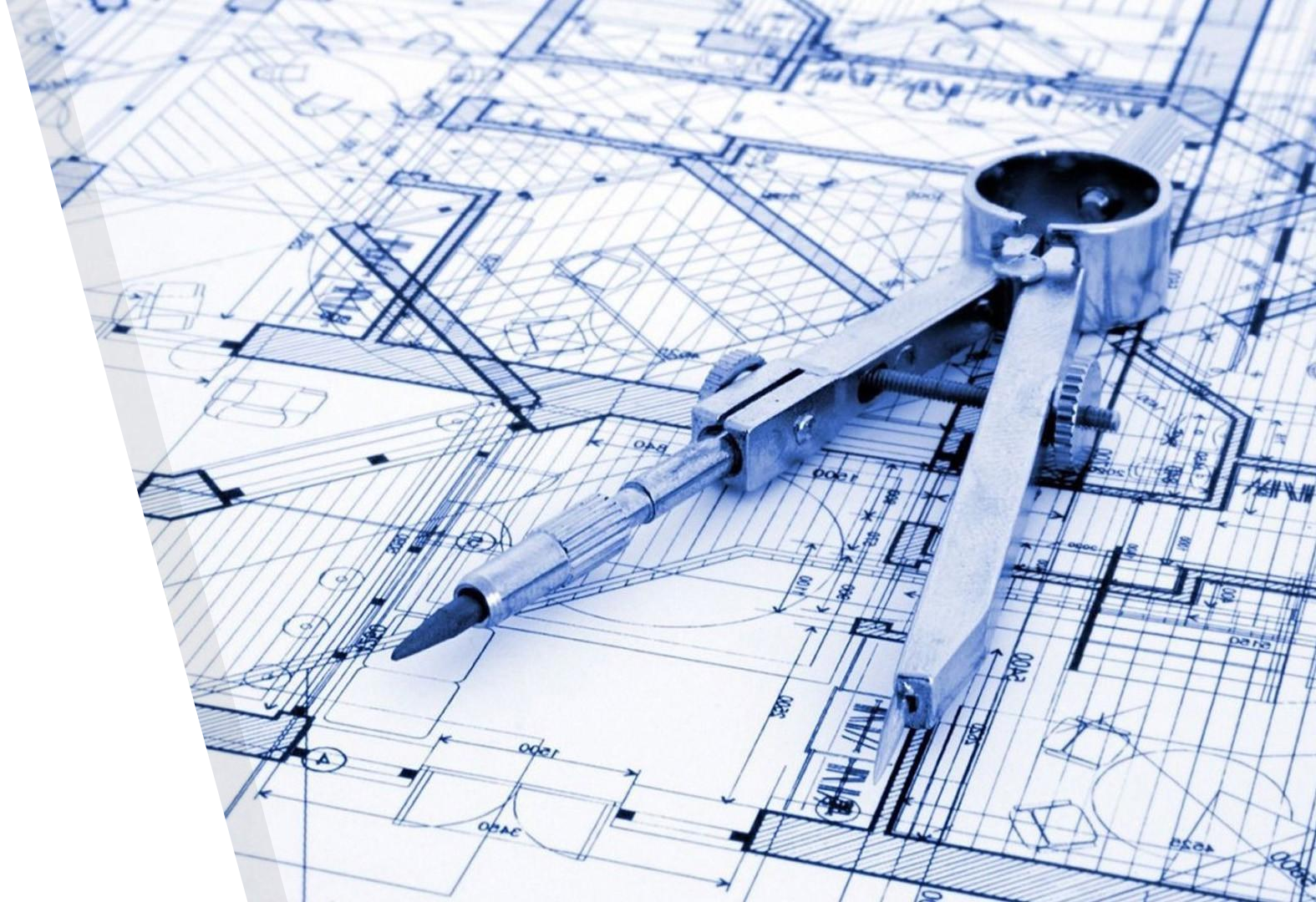
Delta Architecture





1.4

DATA MESH

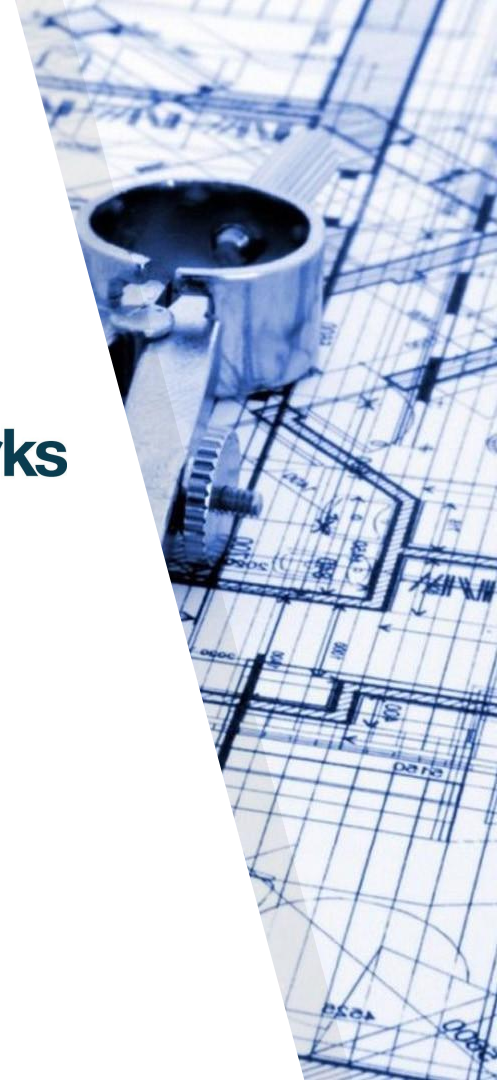


Data Mesh

2019/2020

Proposed by Zhamak Dehghani (Thoughtworks)

It's an **organizational pattern rather than a technical architecture**. Data mesh do not enforces any particular technology stack.

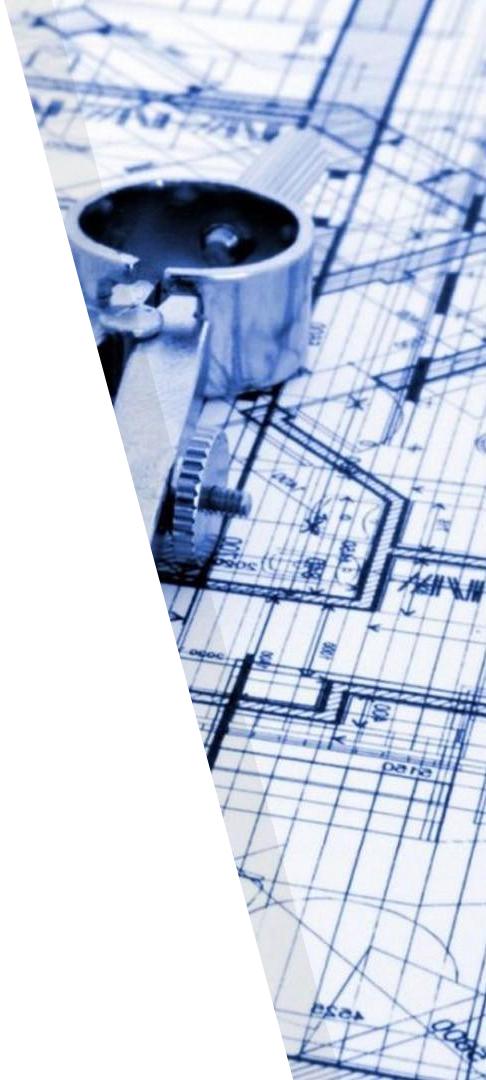


Data Mesh

Data mesh tackles the deficiency of the traditional centralized and monolithic data lake and data warehouse architectures.

These type of architectures are managed by a centralized data teams and sooner than later they became a bottleneck.

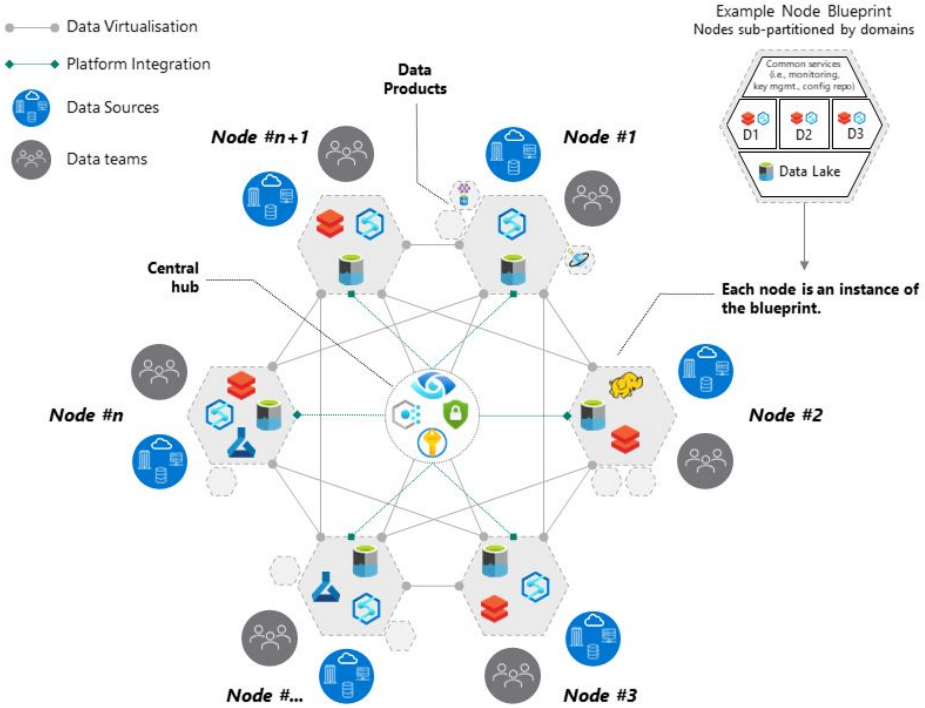
Data Mesh core idea is to distribute the ownership to company domains.



Data Mesh

The Four Principles of Data Mesh

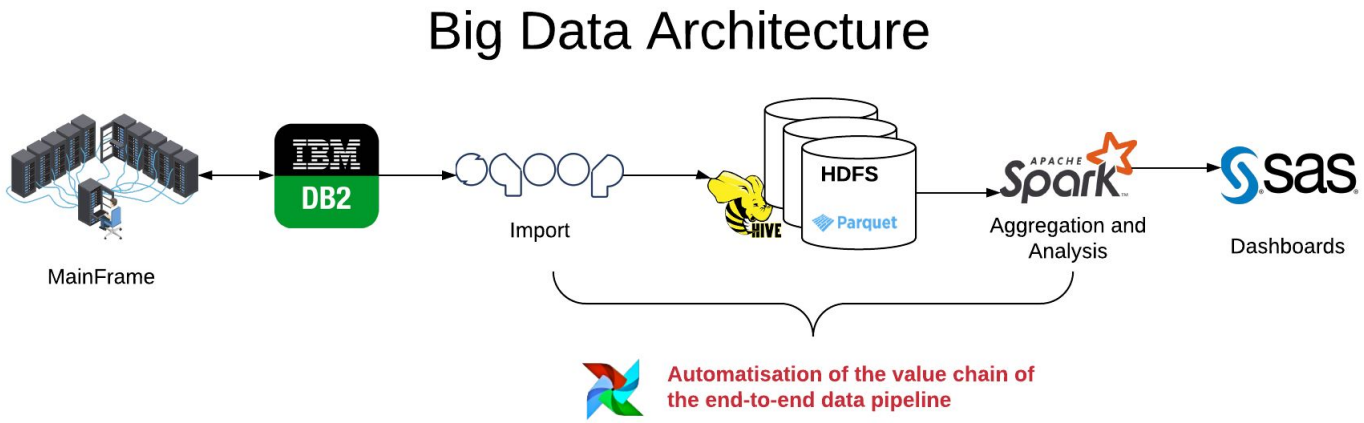
Data Mesh


Data Virtualisation
Platform Integration
Data Sources
Data teams
Data Products
Node #n+1
Node #1
Central hub
Node #n
Node #2
Node #...
Node #3
Example Node Blueprint
Nodes sub-partitioned by domains
Common services (i.e., monitoring, key mgmt., config repo)
D1
D2
D3
Data Lake
Each node is an instance of the blueprint.

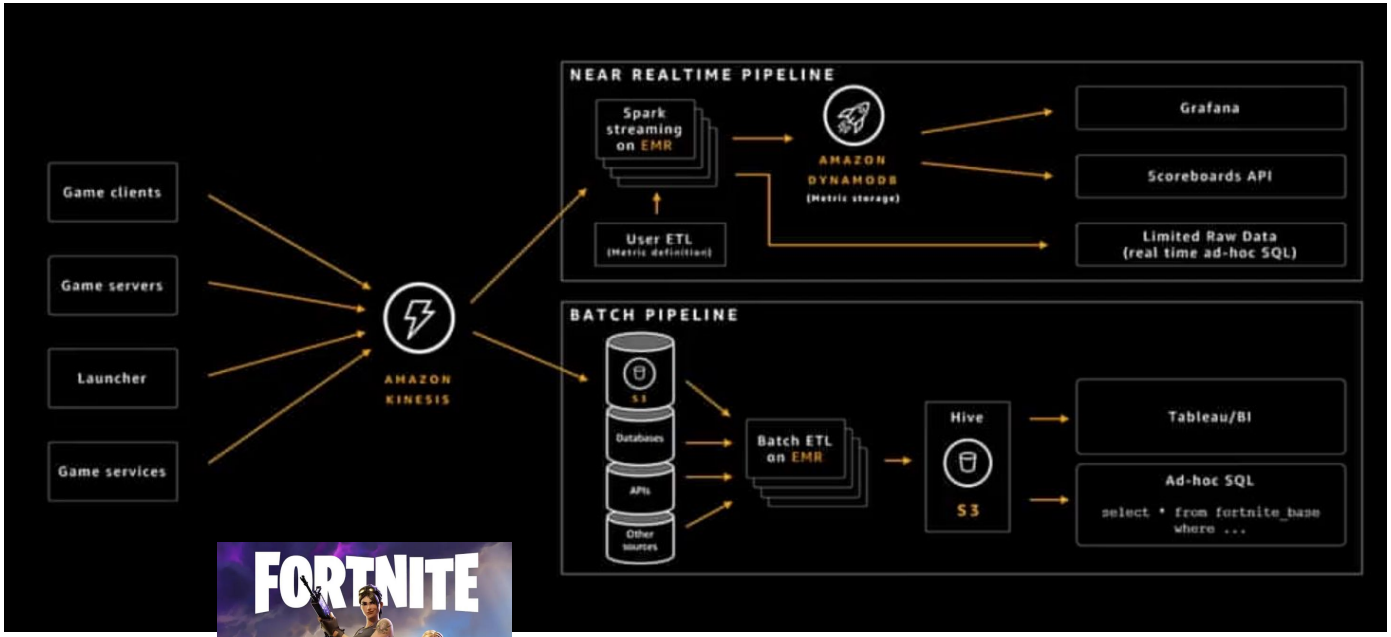
1.5

ARCHITECTURE EXAMPLES

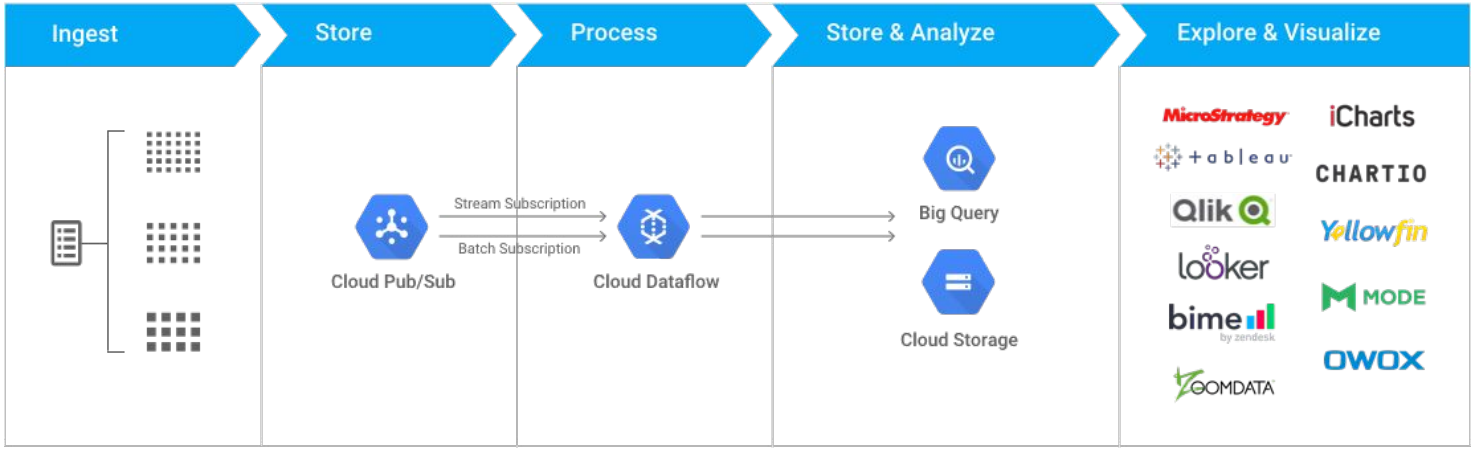
Batch Architecture (On Premise)



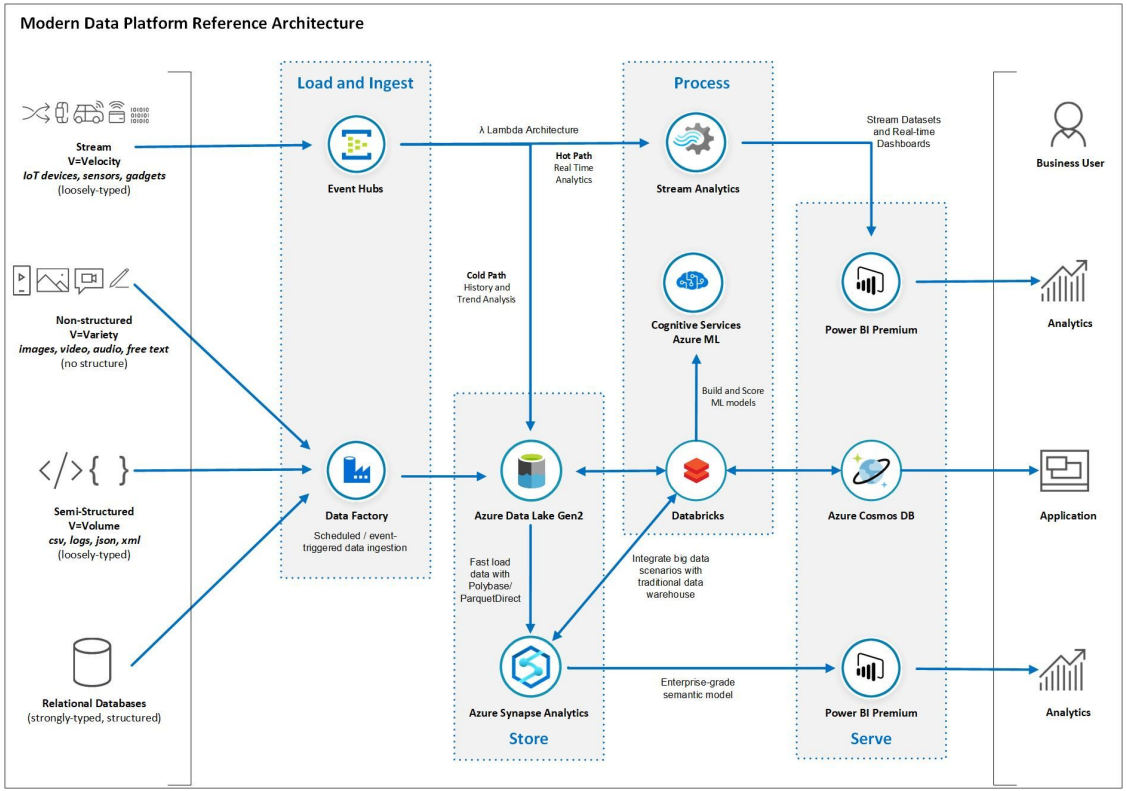
Lambda Architecture (AWS)



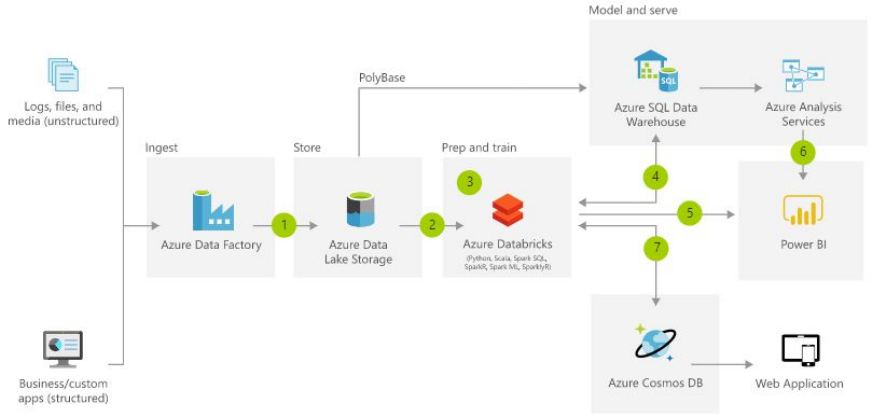
Lambda Architecture (GCP)



Lambda Architecture (Azure)

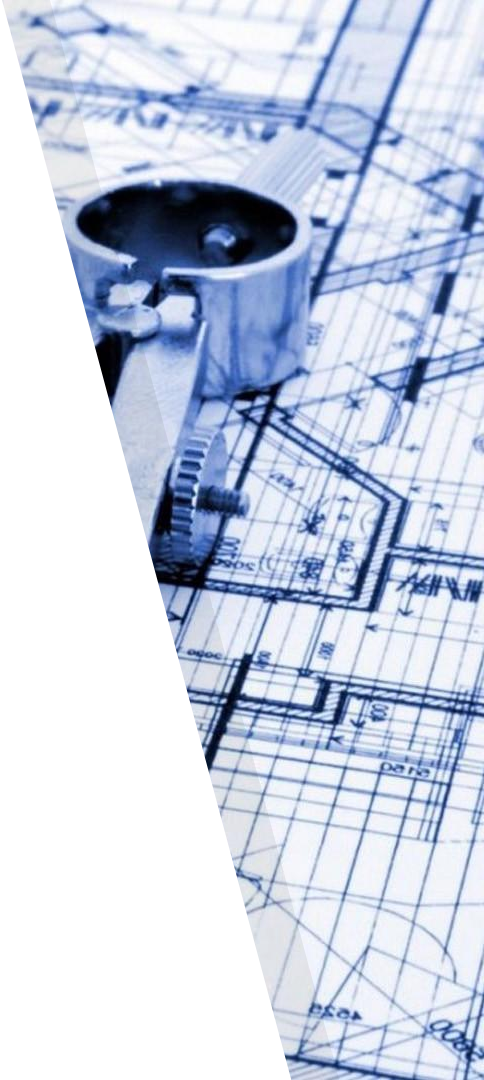


Advanced Analytics Architecture (Azure)

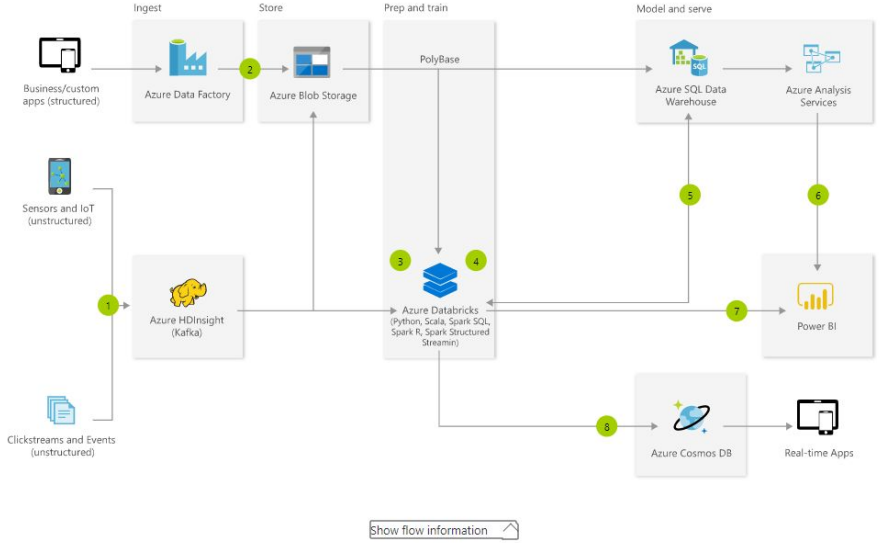


Show flow information

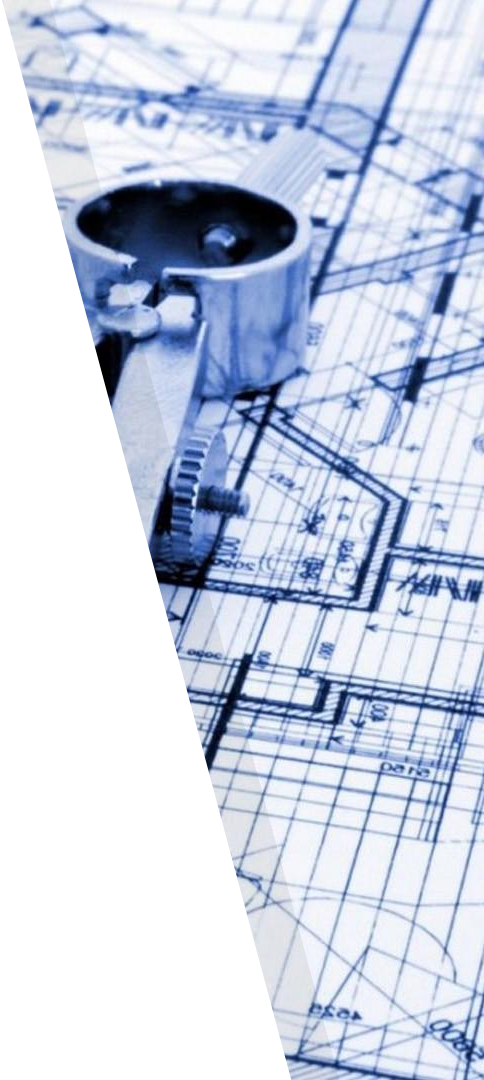
1. Bring together all your structured, unstructured and semi-structured data (logs, files and media) using Azure Data Factory to Azure Blob Storage.
2. Use Azure Databricks to clean and transform the structureless datasets and combine them with structured data from operational databases or data warehouses.
3. Use scalable machine learning/deep learning techniques, to derive deeper insights from this data using Python, R or Scala, with inbuilt notebook experiences in Azure Databricks.
4. Leverage native connectors between Azure Databricks and Azure SQL Data Warehouse to access and move data at scale.
5. Power users take advantage of the inbuilt capabilities of Azure Databricks to perform root cause determination and raw data analysis.
6. Run ad hoc queries directly on data within Azure Databricks.
7. Take the insights from Azure Databricks to Cosmos DB to make them accessible through web and mobile apps.



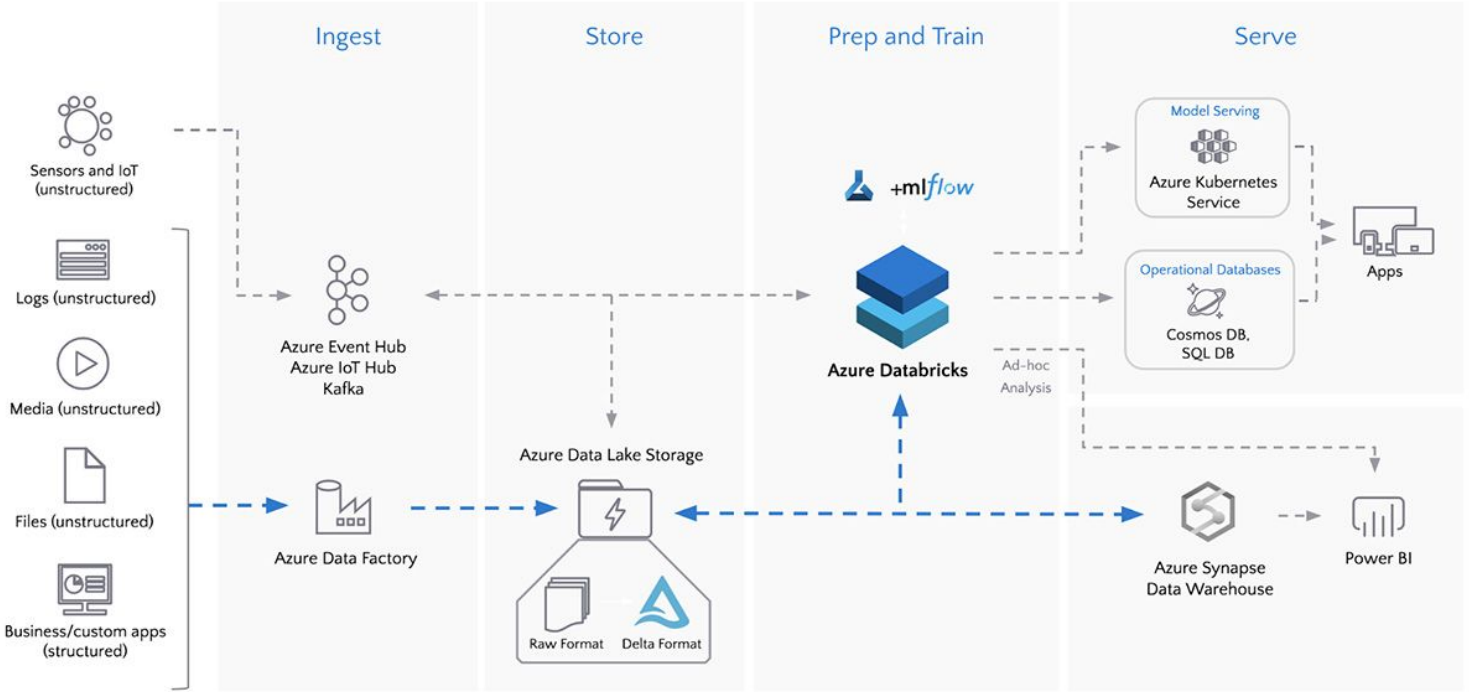
Real-Time Architecture (Azure)



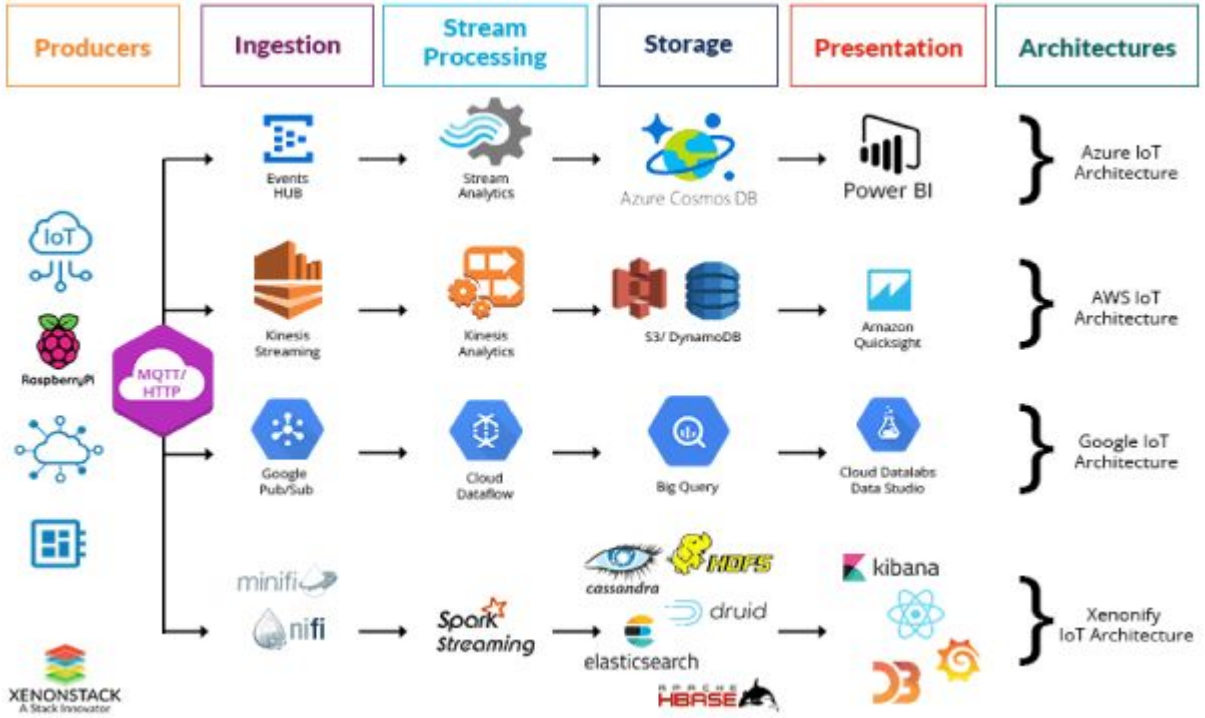
1. Easily ingest live streaming data for an application using Apache Kafka cluster in Azure HDInsight.
2. Bring together all your structured data using Azure Data Factory to Azure Blob Storage.
3. Take advantage of Azure Databricks to clean, transform and analyze the streaming data and combine it with structured data from operational databases or data warehouses.
4. Use scalable machine learning/deep learning techniques, to derive deeper insights from this data using Python, R or Scala, with inbuilt notebook experiences in Azure Databricks.
5. Leverage native connectors between Azure Databricks and Azure SQL Data Warehouse to access and move data at scale.
6. Build analytical dashboards and embedded reports on top of Azure Data Warehouse to share insights within your organization and use Azure Analysis Services to serve this data to thousands of users.
7. Power users take advantage of the inbuilt capabilities of Azure Databricks and Azure HDInsight to perform root cause determination and raw data analysis.
8. Take the insights from Azure Databricks to Cosmos DB to make them accessible through real time apps.



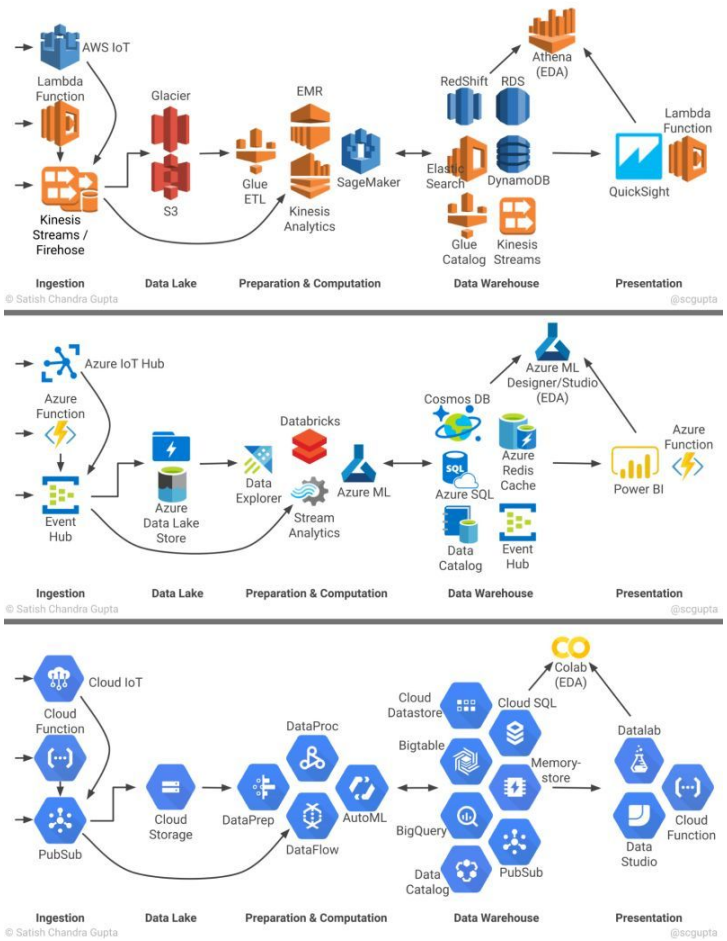
ML/AI Architecture (Azure)



IoT Architectures



Big Data Pipelines on AWS, Microsoft Azure, and GCP

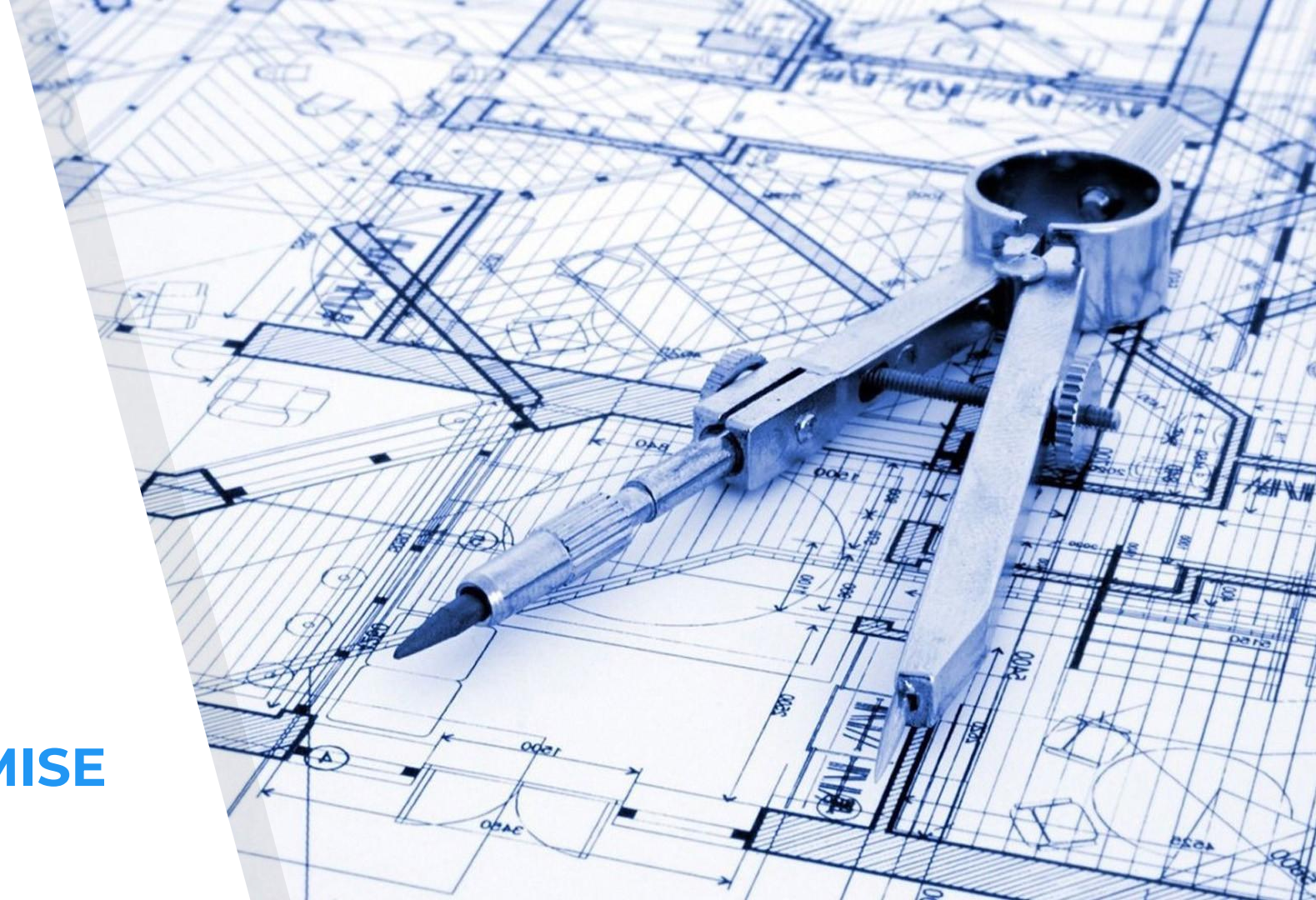


2.

ENTERPRISE DATA ARCHITECTURES

2.1

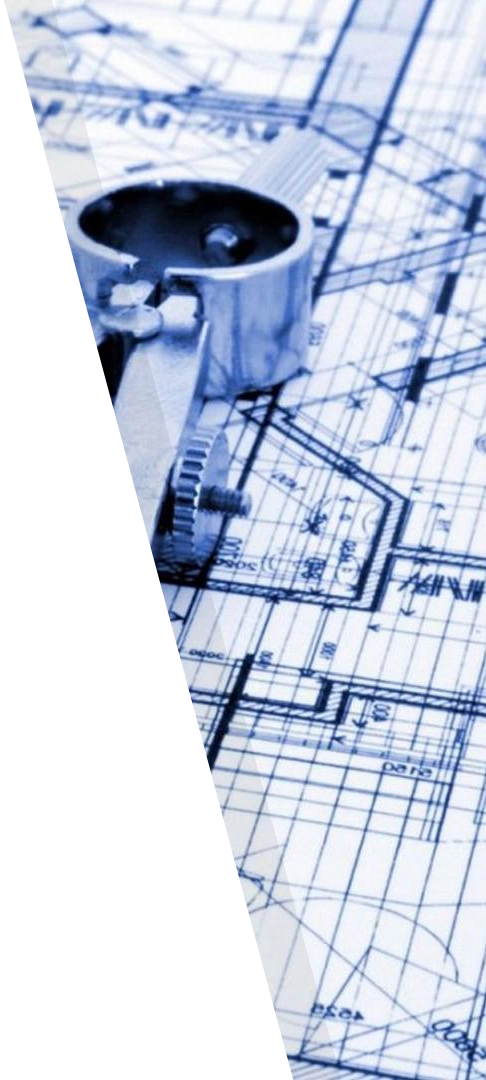
ON PREMISE



On Premise

Resources are **hosted in-house** (enterprise's premises)

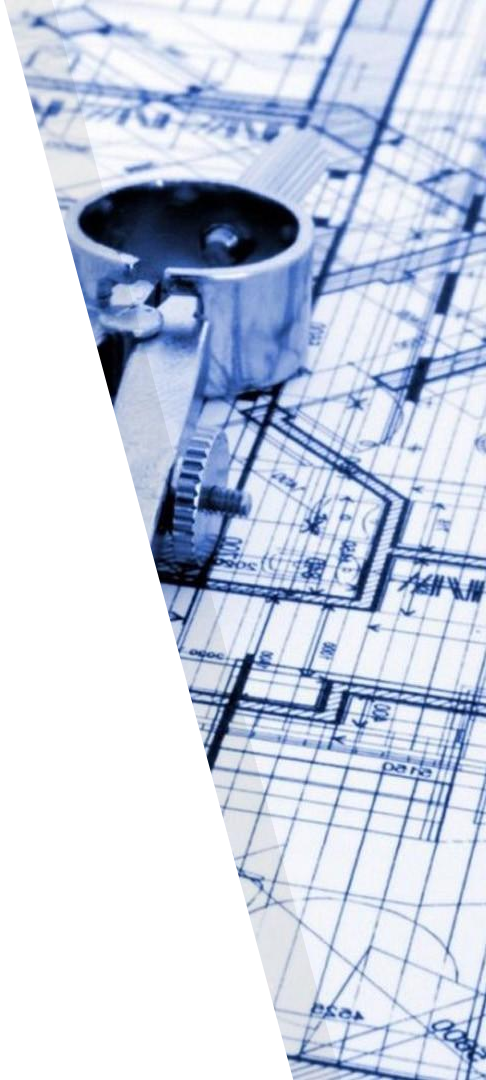
The enterprise is responsible for maintaining the solution and all its related processes.



On Premise

Cost:
Enterprises are responsible for the ongoing costs of the server hardware, power consumption, space and salaries IT teams. **Huge investment upfront**

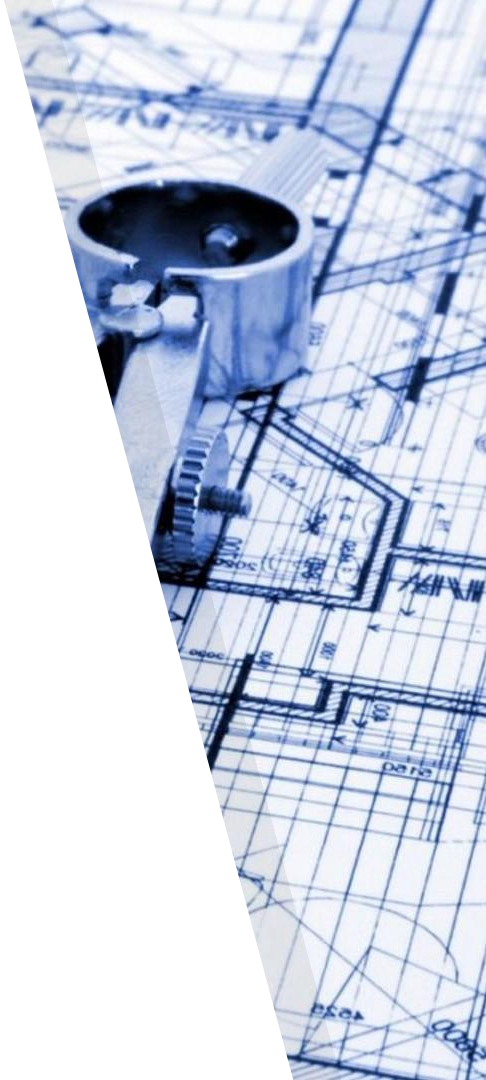
Complexity:
Enterprises have to order the hardware, cable it up, install the software and get the software up an running. Preparing the physical environment alone is hard enough



On Premise

Elasticity:

Growing the infrastructure requires adding new servers and this takes a lot of time (budget, order, shipment, installation, ...)



On Premise

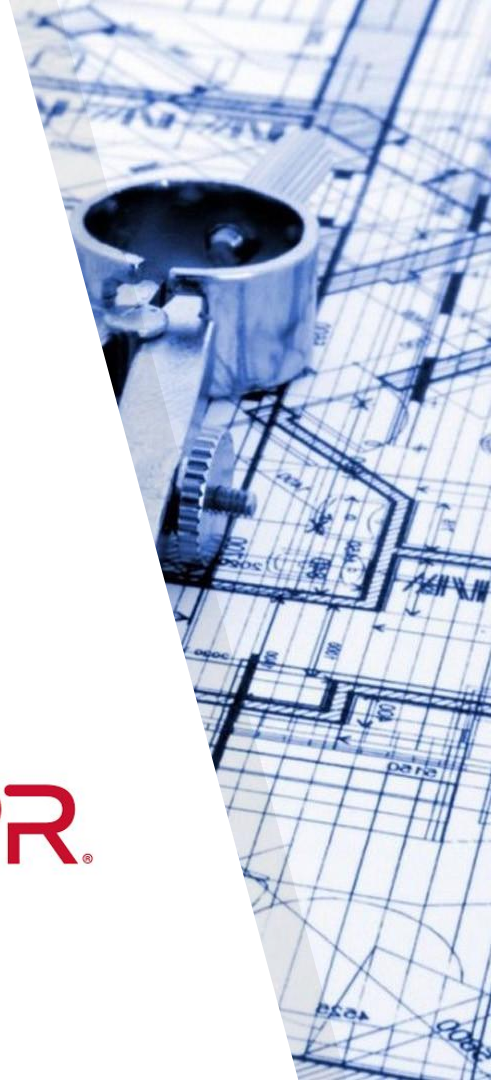
big data distributions:

Big data architectures were deployed on prem with vendor products like these ones:

CLOUDERA

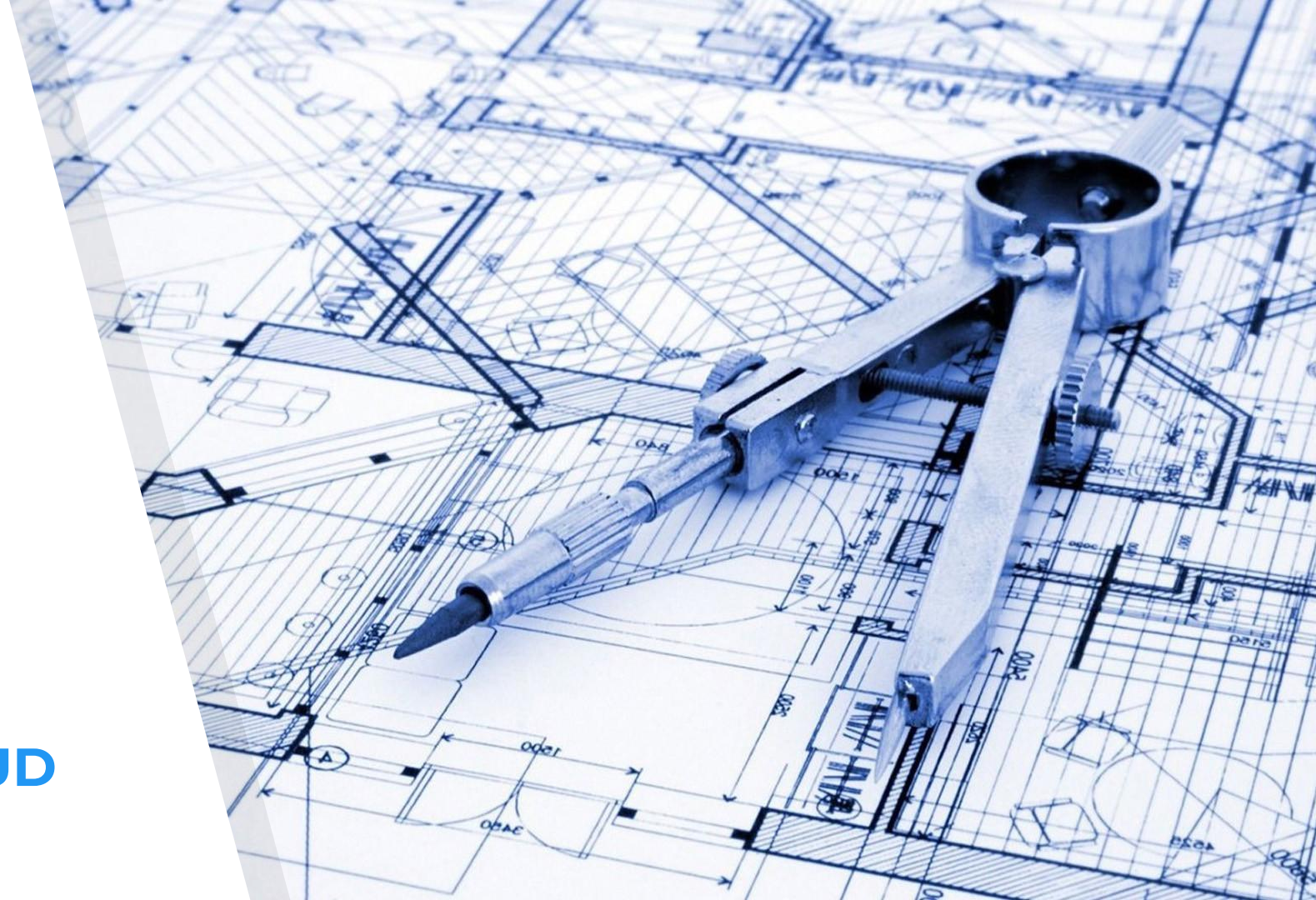
HORTONWORKS®
POWERING THE FUTURE OF DATA™

MAPR®



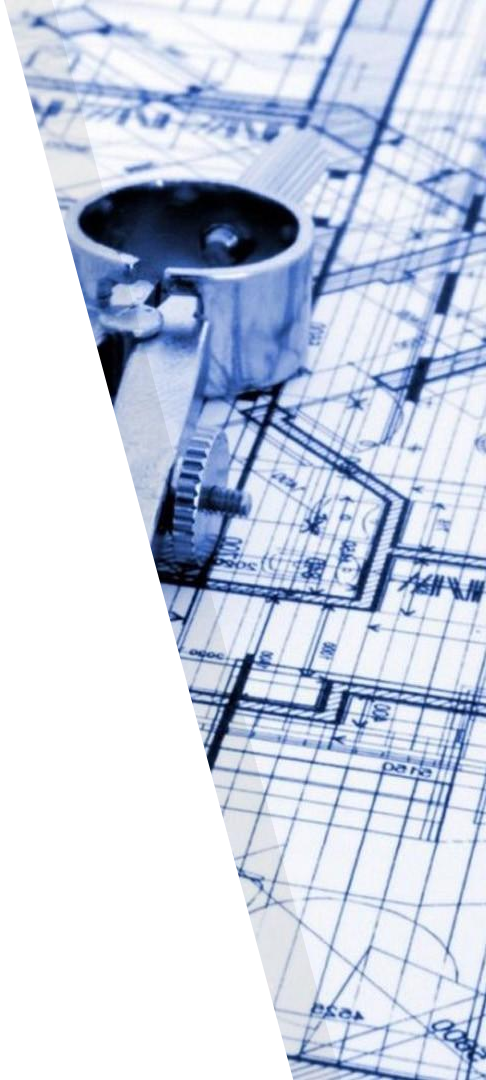
2.2

ON CLOUD



On Cloud

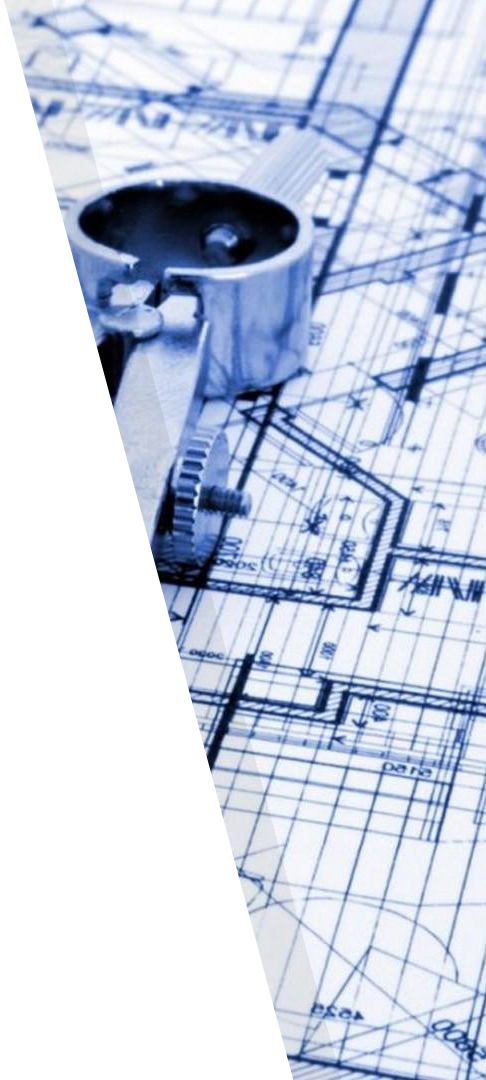
Resources are **hosted on** the premises of the **cloud provider** but enterprises are able to access those resources and use as much as they want at any given time.



On Cloud

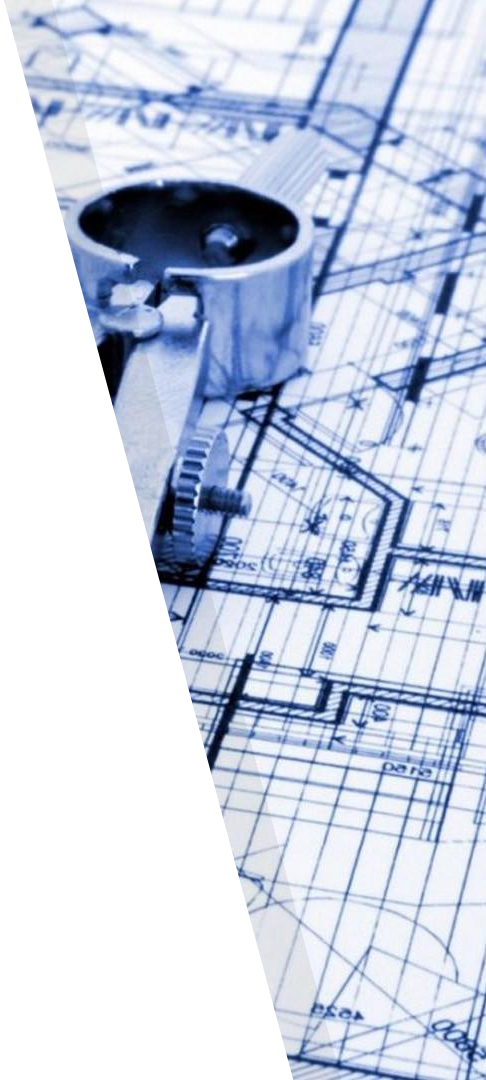
Cost:
Enterprises only **pay for** the resources that they **use**, with none of the maintenance and upkeep costs, and the price adjusts up or down depending on how much is consumed. **Pay as you go model**

Complexity:
Cloud providers automate and **simplify the management** of this technologies.

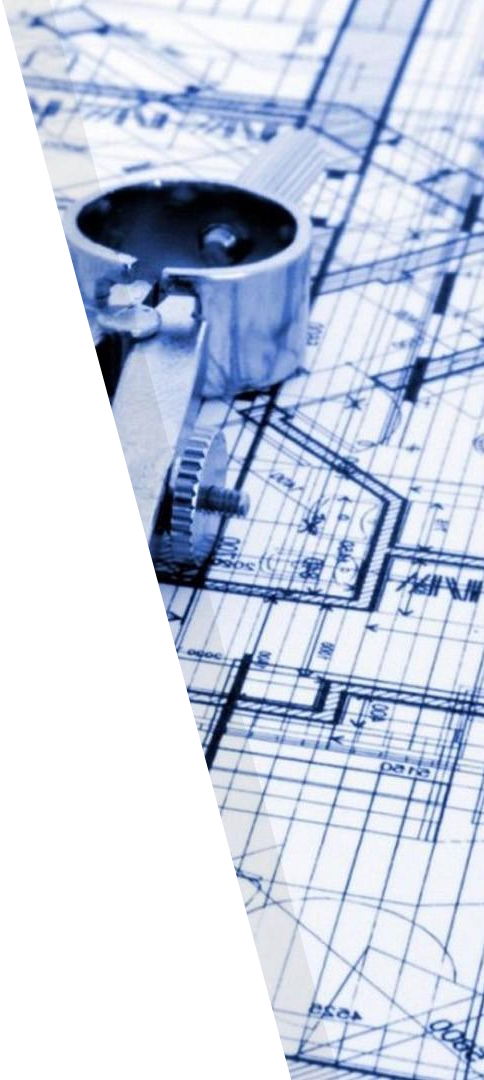


On Cloud

Elasticity:
Enterprises can adapt the amount of resources dynamically according to their needs or current budget.



On Cloud




EC2


Auto Scaling


Lambda


Key Pair


IAM


Cost Explorer


S3

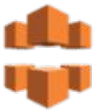

EBS


Snapshot


DynamoDB


RDS


CloudWatch


CloudFront


Route 53


VPC


Elastic IP

Route Table


SQS


ACL


ELB


NAT Gateway


Internet Gateway


Security Group

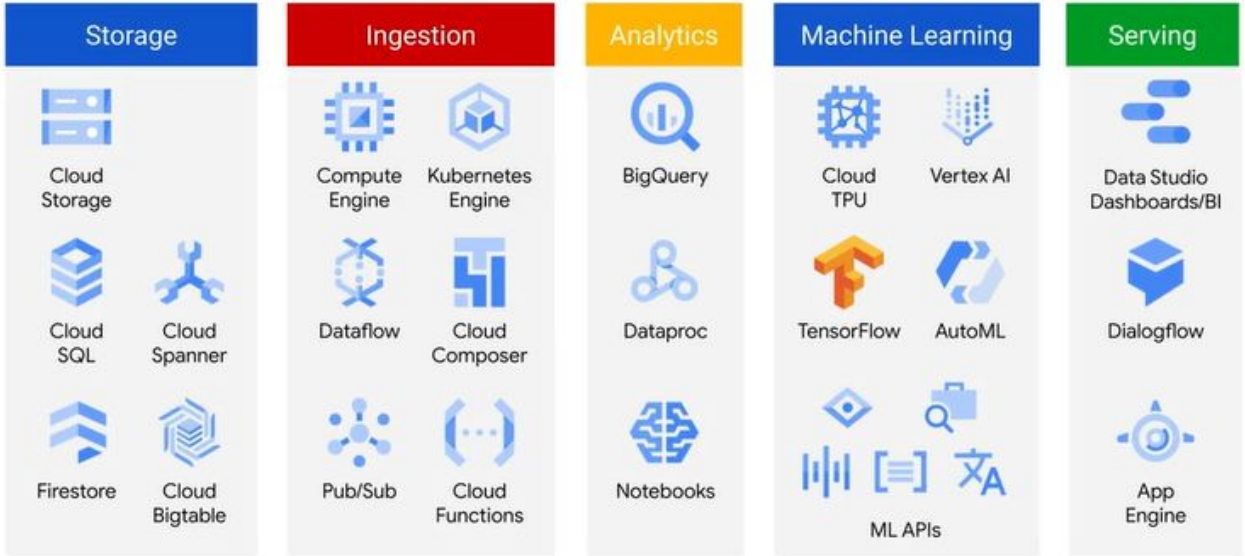

SNS

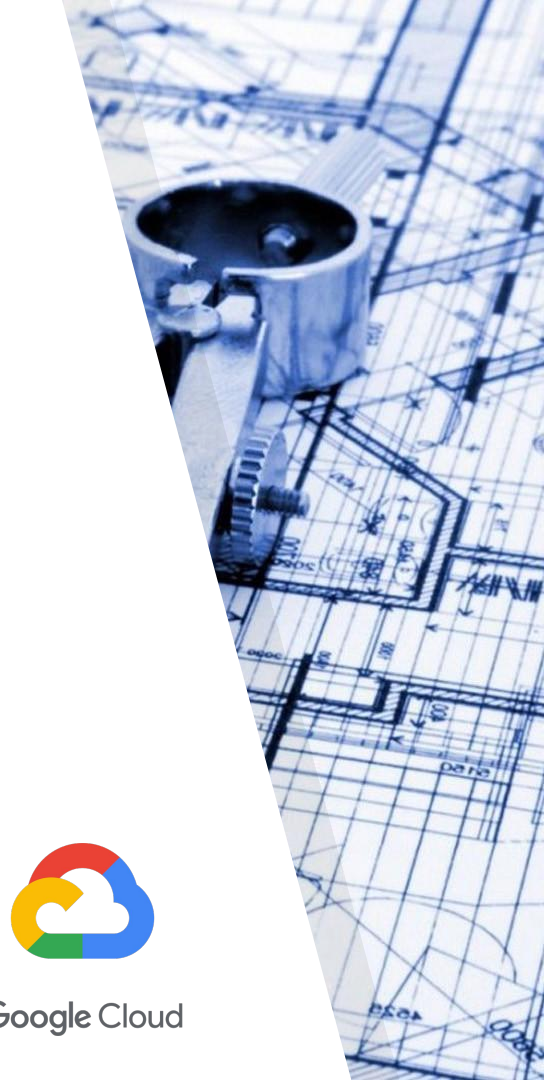
On Cloud

Azure

The Azure **big** data landscape

On Cloud





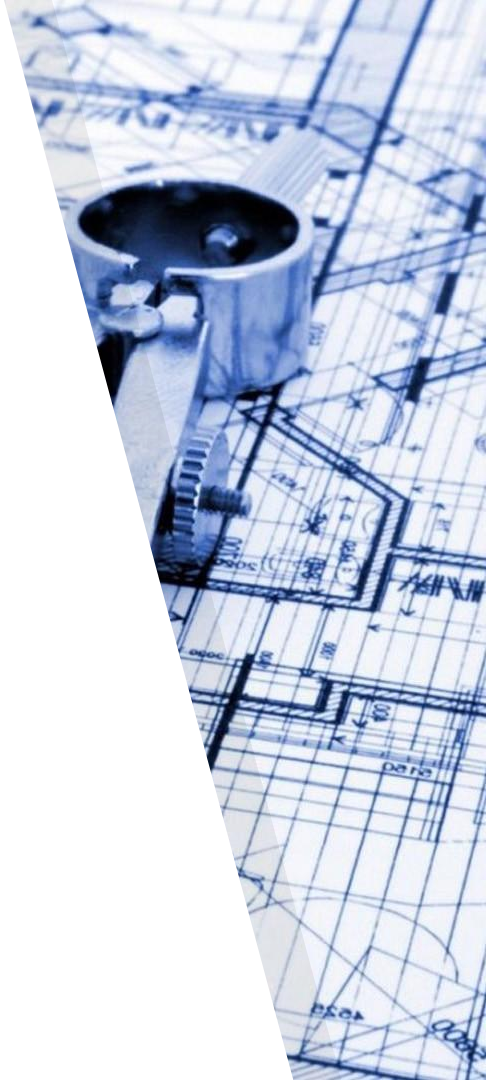
Google Cloud

2.3

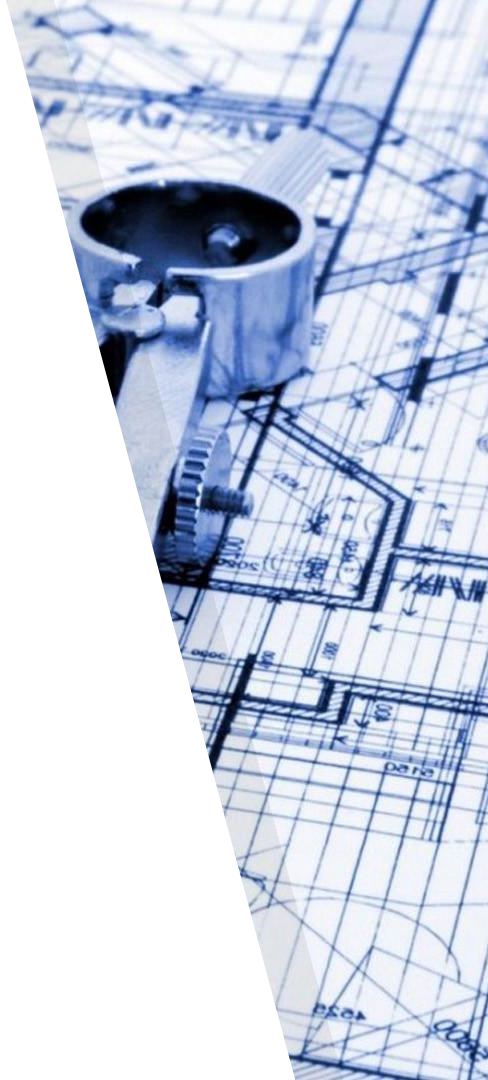
HYBRID

Hybrid

Enterprises that have extra **sensitive information**, such as **government and banking** must have a certain level of **security and privacy** that only **on-premises** environment provides.

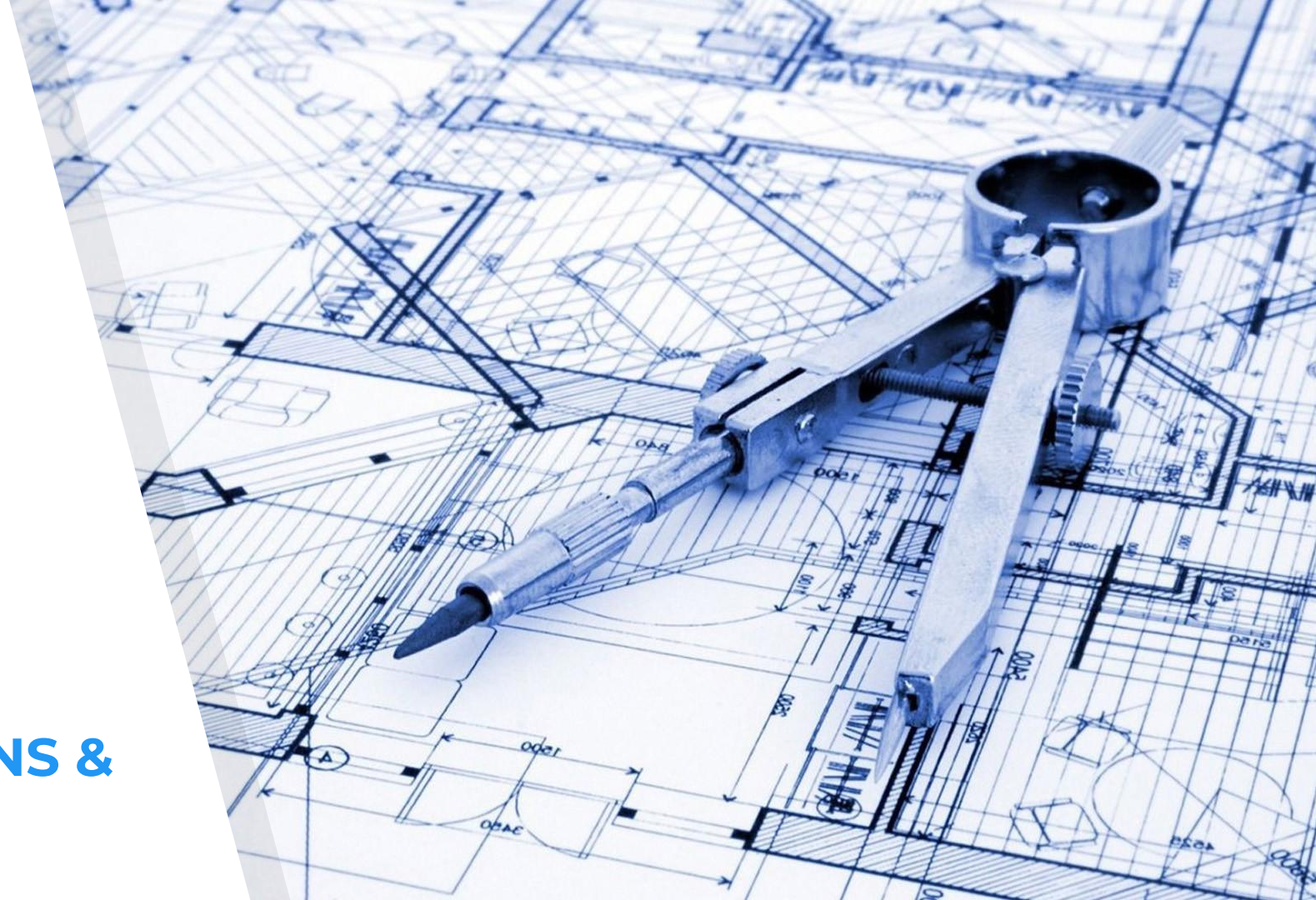


Hybrid



3.

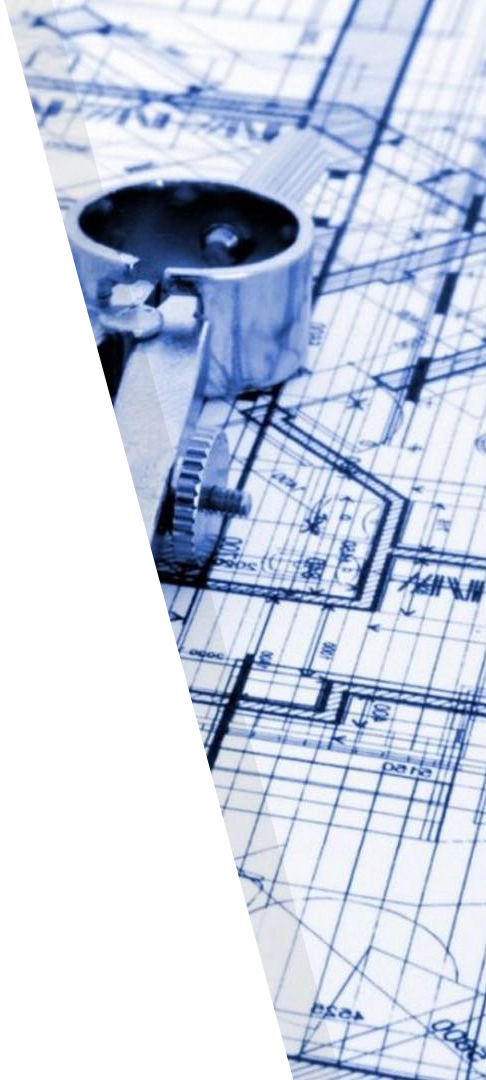
DATA POSITIONS & ROLES



Roles

There are four main job roles related to big data ecosystem:

- Data Analysts
- Data Scientists
- Data Engineers
- Data Architects

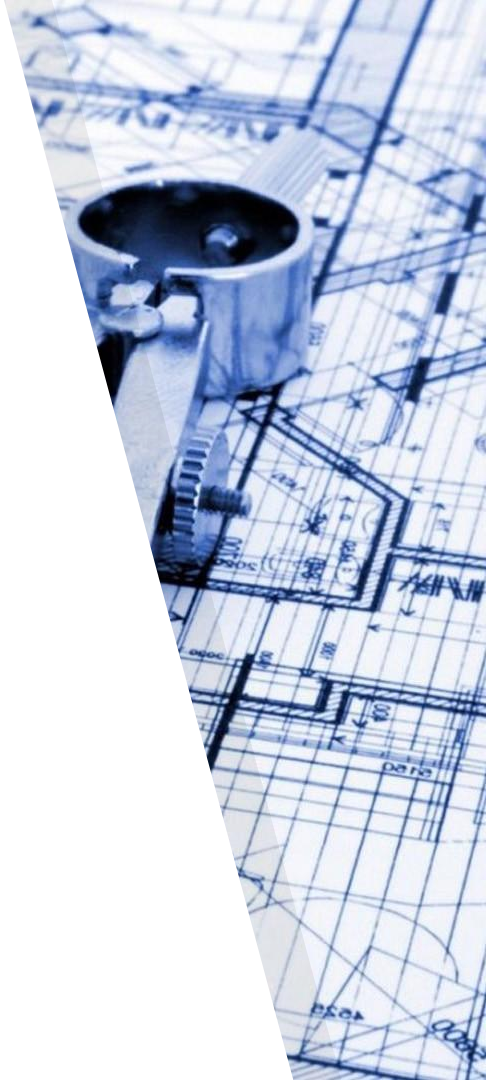


Roles - Data Analyst

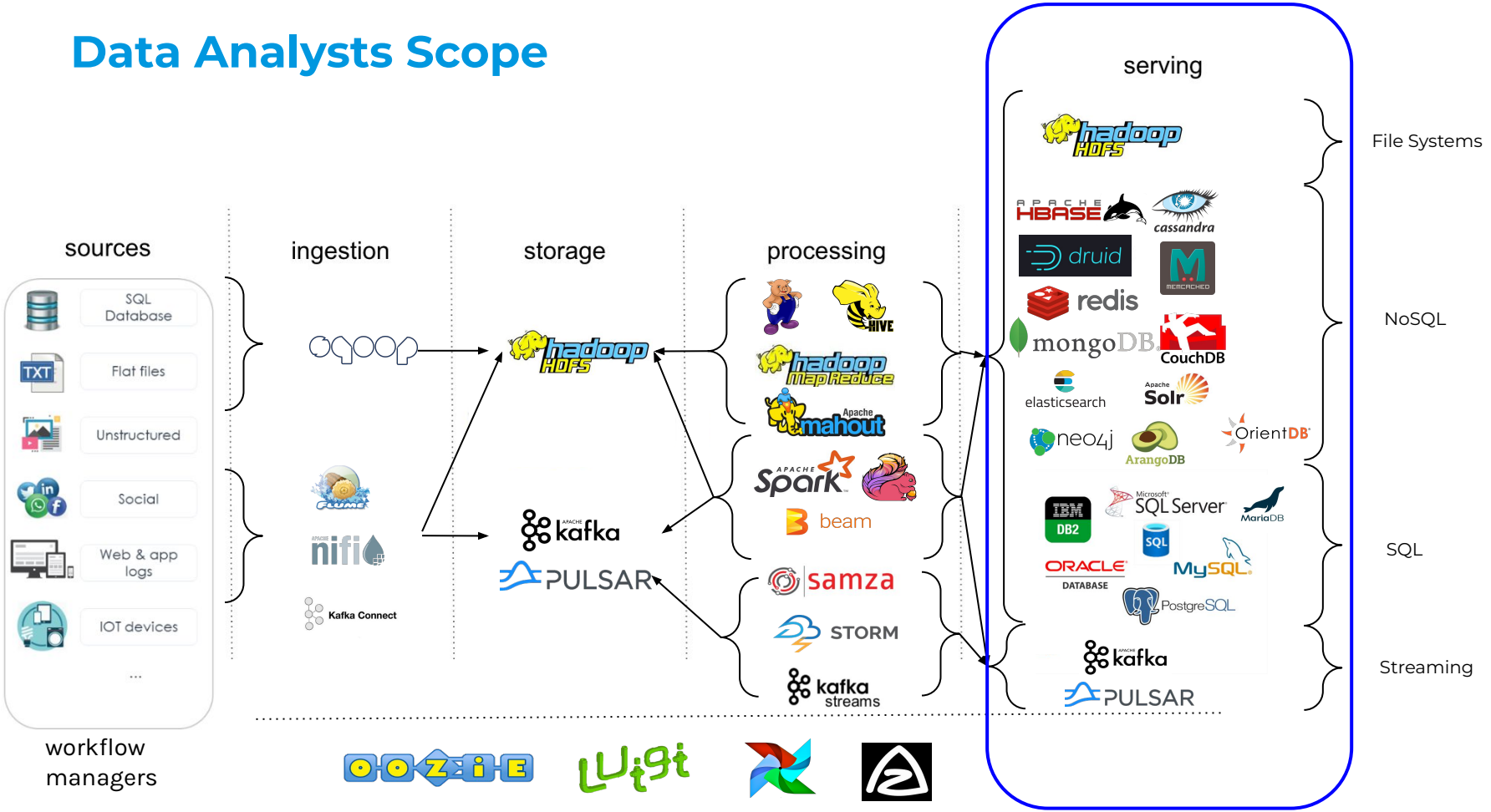
Data Analyst

(aka Business Analyst) Are experts in the domain/business they work in.

They analyze the data to provide solutions that enhance business processes based on their analysis.



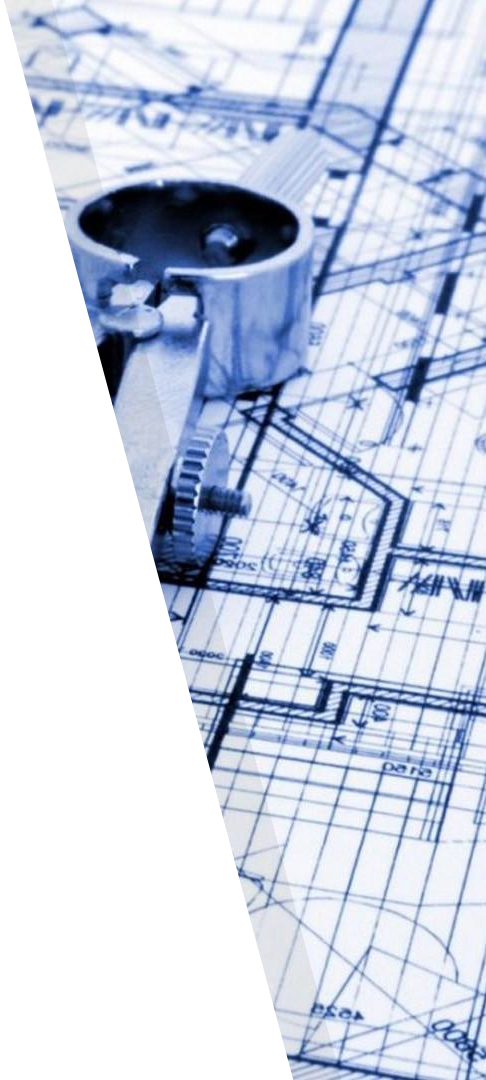
Data Analysts Scope



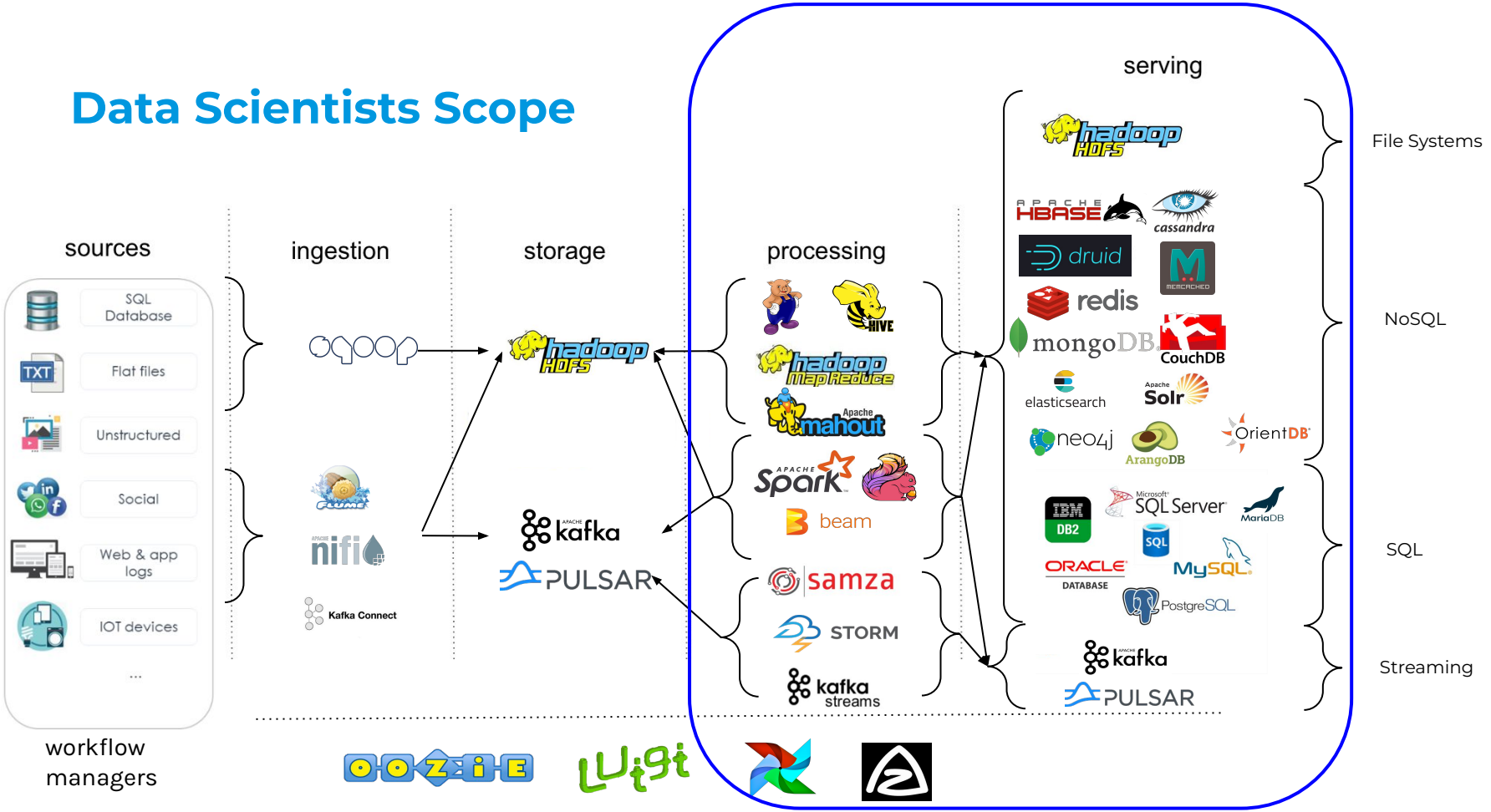
Roles - Data Scientist

Data Scientist

Data scientists are a new breed of analytical data expert who have the technical skills to solve complex problems using machine learning and AI.



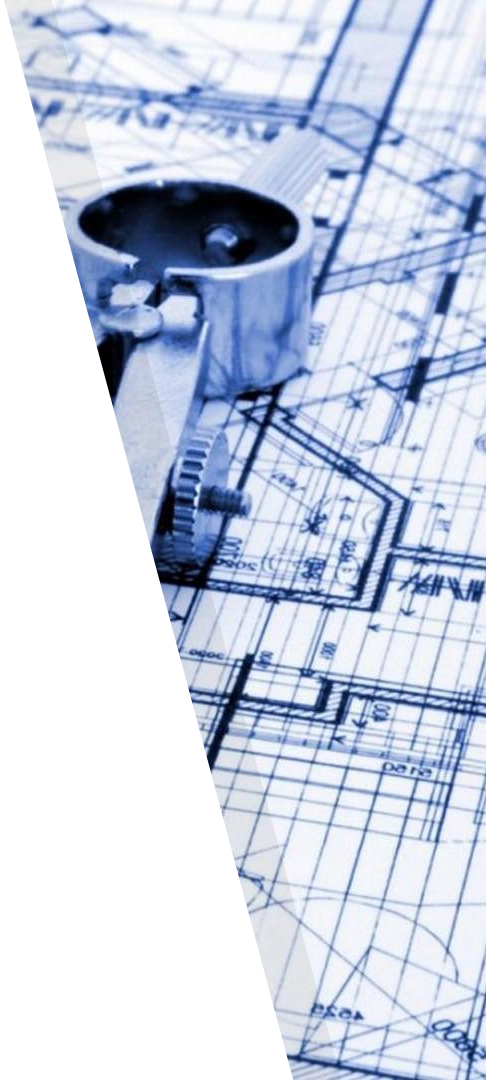
Data Scientists Scope



Roles - Data Engineer/Architect

Data Engineer/Architect

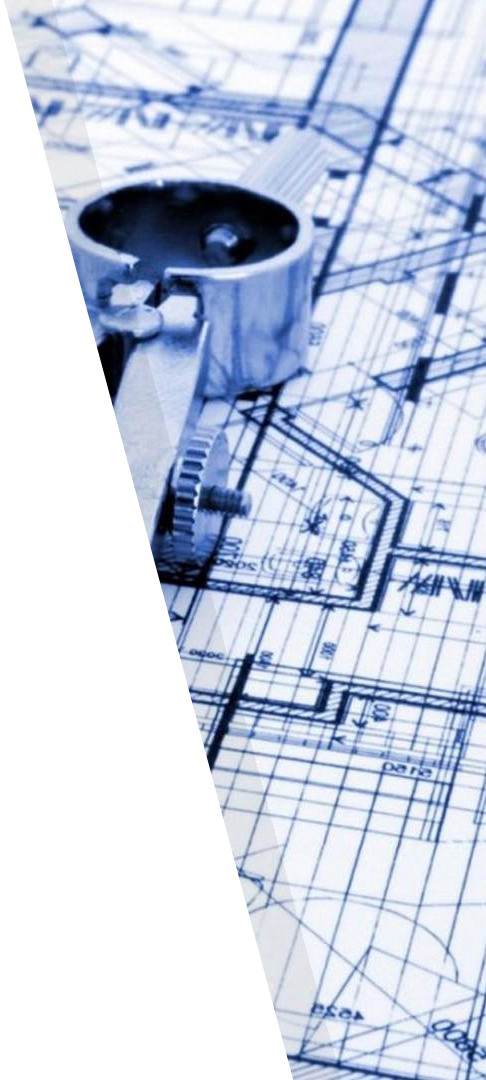
Data engineers and architects are the designers, builders and managers of the big data infrastructure/technologies.



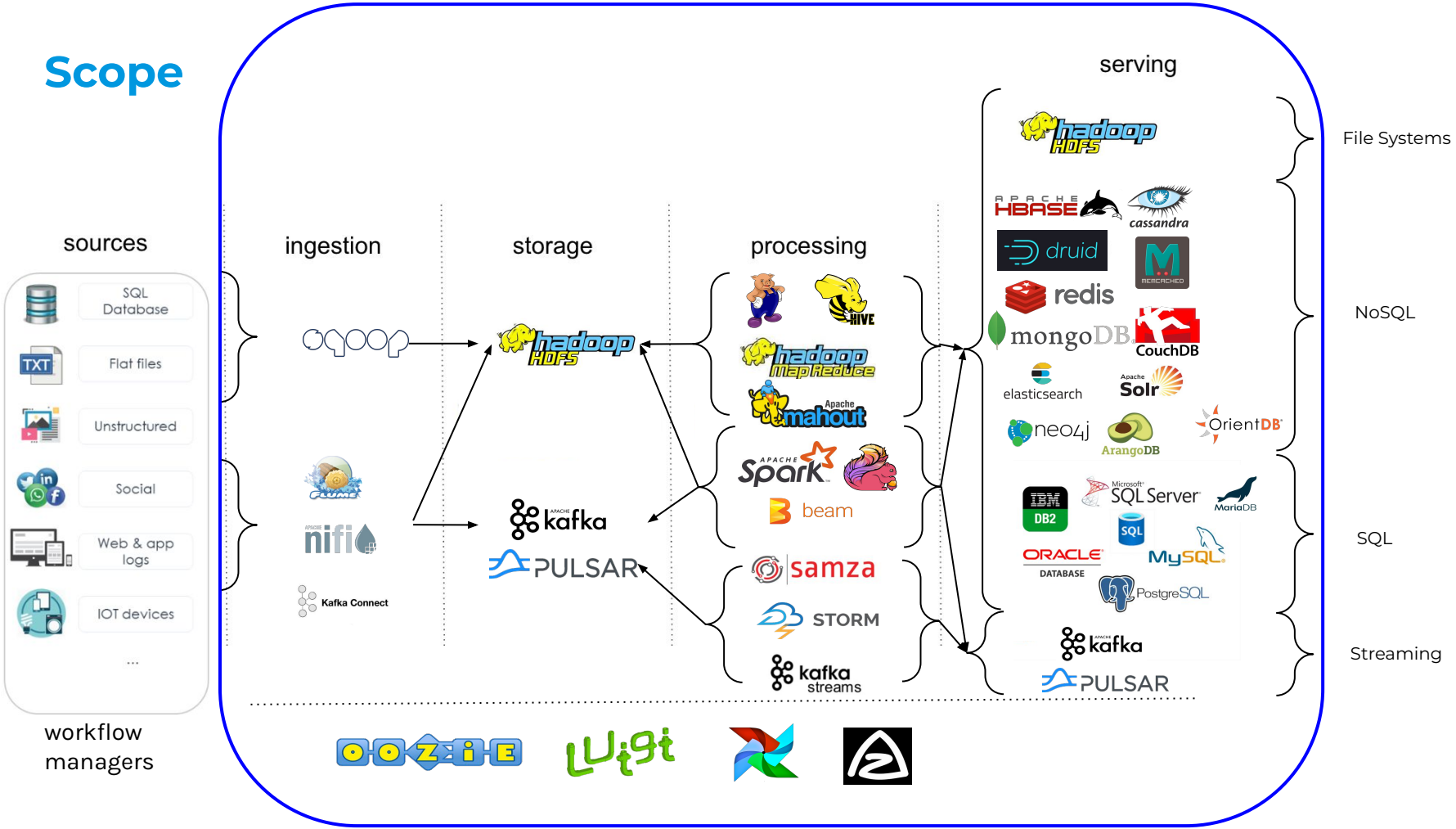
Roles - Data Engineer/Architect

Data Engineer/Architect

They ensure that an organization’s data architecture is running without problems for data scientists and data analysts to carry out their analysis.

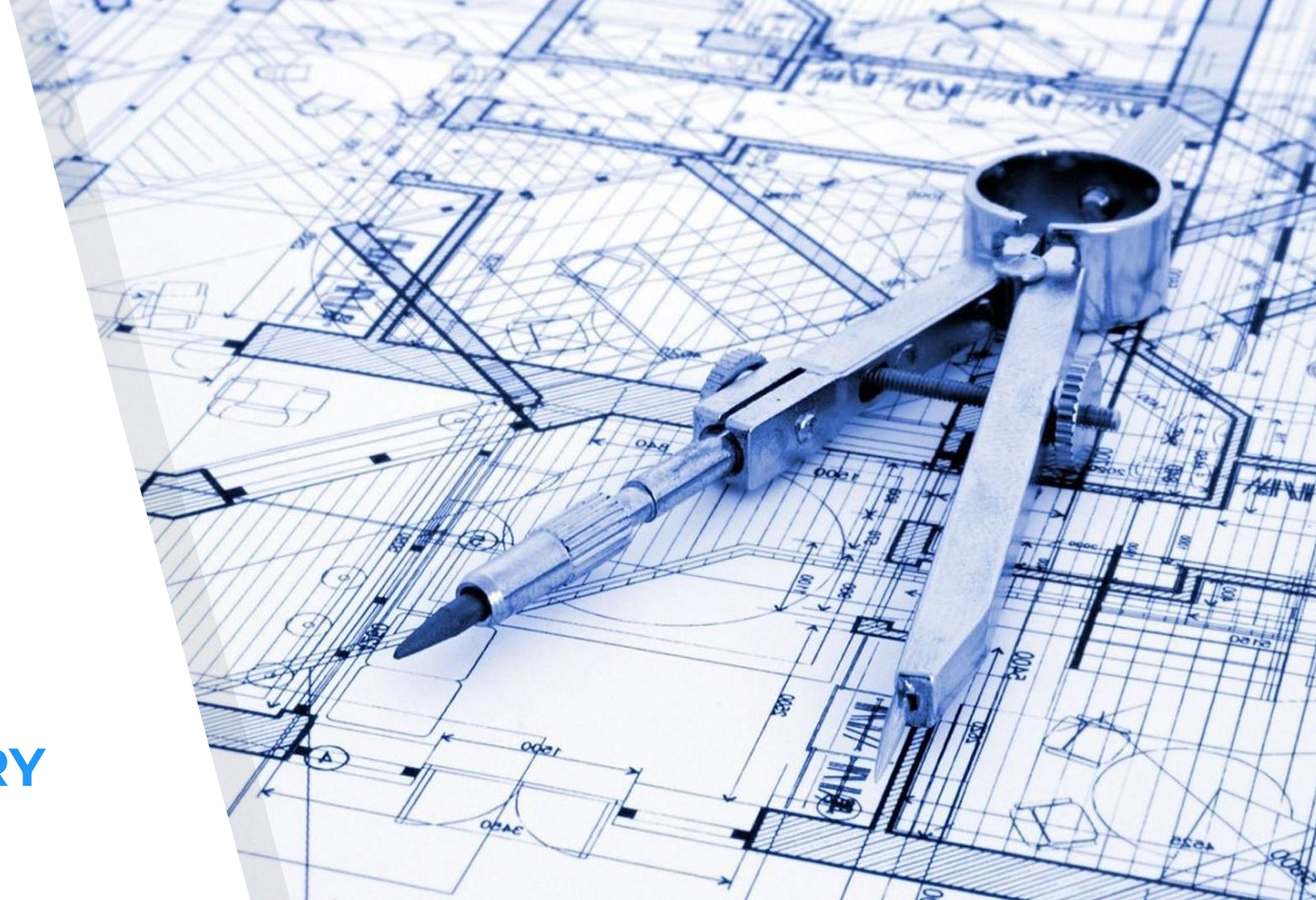


Scope


sources
SQL Database
Flat files
Unstructured
Social
Web & app logs
IOT devices
...
workflow managers
ingestion
storage
processing
serving
Kafka Connect
hadoop HDFS
kafka
PULSAR
HIVE
hadoop MapReduce
mahout
Spark
beam
samza
STORM
kafka streams
HBASE
cassandra
druid
redis
mongoDB
CouchDB
elasticsearch
Solr
neo4j
ArangoDB
OrientDB
DB2
SQL Server
MariaDB
ORACLE DATABASE
MySQL
PostgreSQL
OOZIE
Luigi
File Systems
NoSQL
SQL
Streaming

4.

SUMMARY

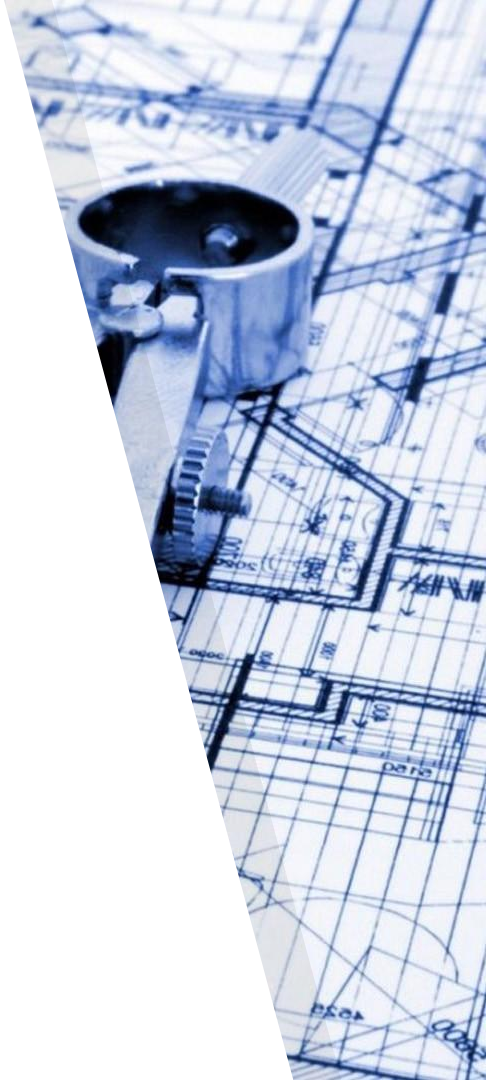


Summary

There are different data architectures patterns (**lambda, kappa, delta**) that support many different data pipelines and scenarios

Modern data architectures are cloud based.

Clouds provide many technologies (**open source** & **proprietary**) to build any type of data architecture.



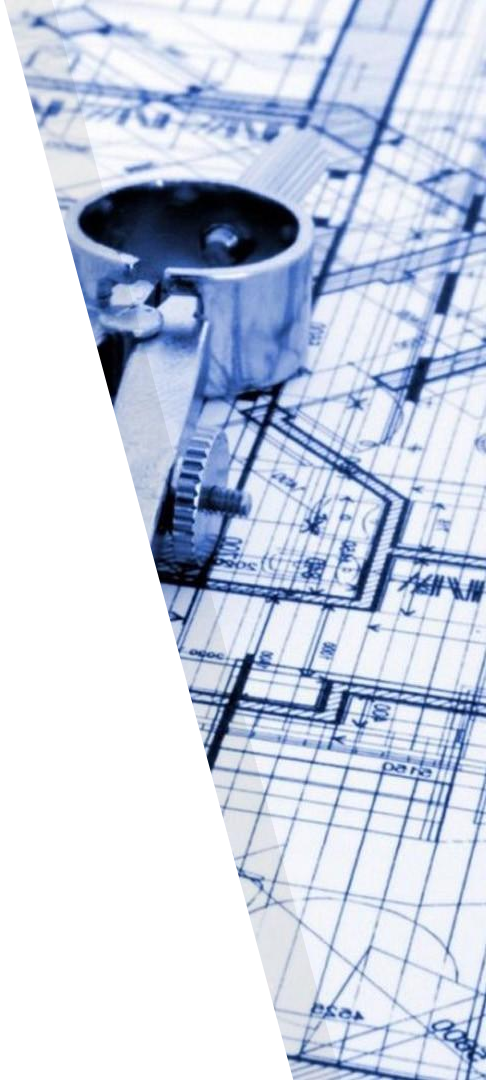
Interesting Reads about Data Mesh

Data Mesh Principles

https://martinfowler.com/articles/data-mesh-principles.html

The four principles of Data Mesh

https://www.thoughtworks.com/en-es/about-us/events/webinars/core-principles-of-data-mesh



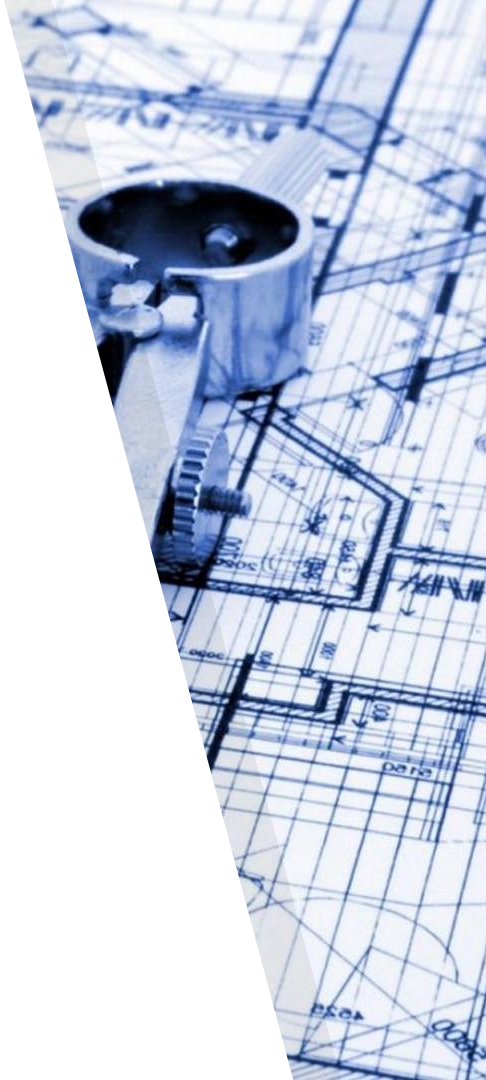
Interesting Reads about Data Mesh

Data Mesh Intro

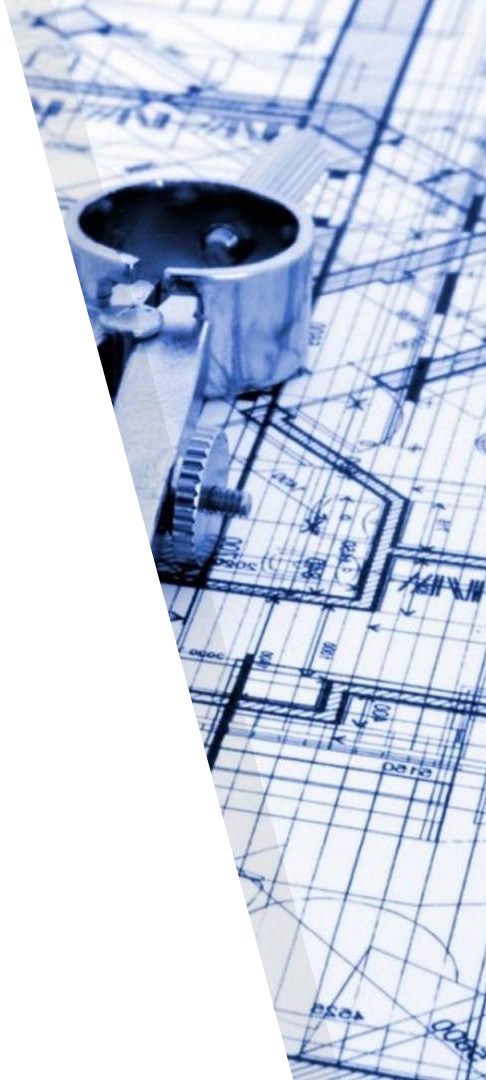
https://developer.confluent.io/learn-kafka/data-mesh/intro/

Building a Data Mesh Architecture with AWS Lake Formation

https://www.youtube.com/watch?v=YPYODx4Pfdc



Interesting Reads about Delta



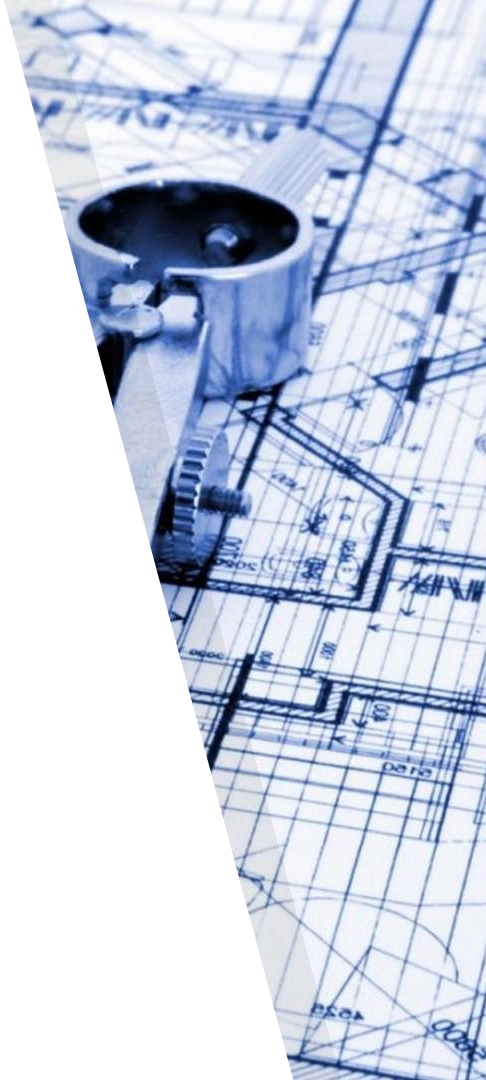
Interesting Reads about Lambda

How to beat the CAP theorem

http://nathanmarz.com/blog/how-to-beat-the-cap-theorem.html

Applying the Lambda Architecture with Spark

https://databricks.com/session/applying-the-lambda-architecture-with-spark



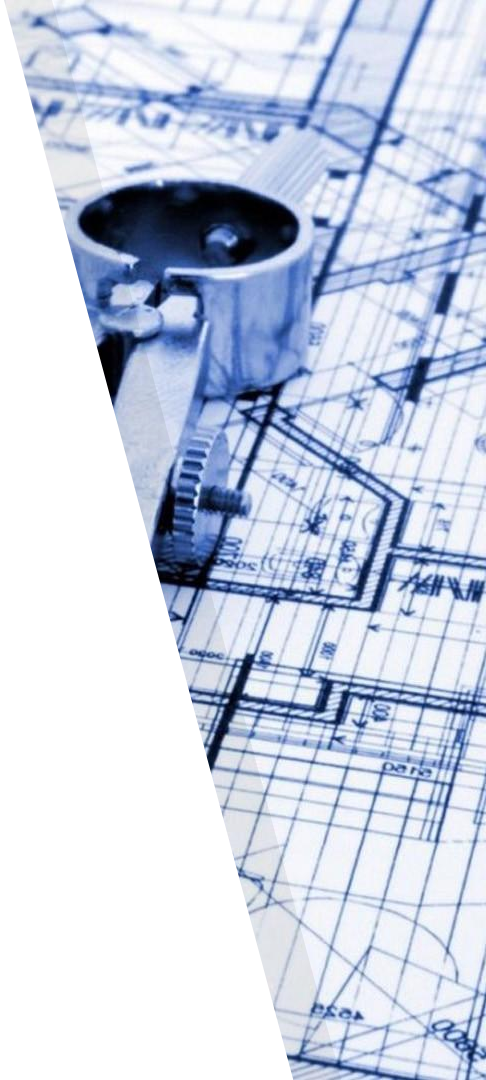
Interesting Reads about Kappa

Questioning the Lambda architecture

http://radar.oreilly.com/2014/07/questioning-the-lambda-architecture.html

Key Big Data Architectures

https://www.youtube.com/watch?v=4haHum0b7Ls



Interesting Reads about Cloud Architectures

Cloud Big Data Architectures

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

https://docs.microsoft.com/en-us/azure/architecture/example-scenario/dataplate2e/data-platform-end-to-end

https://cloud.google.com/products/big-data/