Big Data Storage Solutions

File systems, block access, object storage

Machine Learning + Big Data in Real Time + Cloud Technologies

=> The Future of Intelligent Systems

Where to Find The Code and Materials?

https://github.com/iproduct/course-ml

Agenda for This Lesson - I

- Web scale systems, data lakes and Lambda, Kappa and Zeta architectures.
- Cathegores of solutions for big data storage – file systems, block access, object storage
- Comparison between different types of solutions – storage capacity, bandwidth, latency.
- Distributed file systems for big data storage.
- Introduction to Apache Hadoop, MapReduce/YARN and HDFS.
- Basic HDFS components: Name Node, Secondary Name Node, Job tracker, Data Node, Task Tracker.
- Installing Hadoop and HDFS.

Agenda for This Lesson - II

- Implementing word count demo using Hadoop YARN and HDFS.
- Object storages for big data - Ceph, MinIO, OpenIO. Advantages.
- Installing MiniIO in Windows and as Docker image.
- Testing the install using MinIO Browser and MinIO Client.
- MinIO Python library for Amazon S3 Compatible Cloud Storage – uploading files using Python.

Introduction to Databases and Distributed Data Processing

Databases, DBMSs and DB Models

- Database - an organized collection of data, generally stored and accessed electronically from a computer system. Can be developed using formal design and modeling techniques.
- DataBase Management System (DBMS) – software that interacts with end users, applications, and the database to capture and analyze the data, providing core facilities to create and administer databases.
- DBMSs can be classified according to the database models that they support:
  - In 1980s relational databases became dominant, modelling data as rows and columns in a series of tables, and the vast majority use Structured Query Language (SQL) for writing and querying data.
  - In the 2000s, non-relational databases became popular, referred to as NoSQL because they use different query languages.

Relational Databases

- "Relational database" term – invented by E. F. Codd at IBM in 1970, paper: "A Relational Model of Data for Large Shared Data Banks".
- Present the data to the user as relations (a presentation in tabular form, i.e. as a collection of tables with each table consisting of a set of rows and columns)
- Provide relational operators to manipulate the data in tabular form.
- As of 2009, most commercial relational DBMSs employ SQL as their query language.

```
dvdrental=# select title, release_year, length, replacement_cost from film
dvdrental-#   where length > 120 and replacement_cost > 29.50
dvdrental-#   order by title desc;
         title          | release_year | length | replacement_cost
------------------------+--------------+--------+------------------
 West Lion              |         2006 |    159 |            29.99
 Virgin Daisy           |         2006 |    179 |            29.99
 Uncut Suicides         |         2006 |    172 |            29.99
 Tracy Cider            |         2006 |    142 |            29.99
 Song Hedwig            |         2006 |    165 |            29.99
 Slacker Liaisons       |         2006 |    179 |            29.99
 Sassy Packer           |         2006 |    154 |            29.99
 River Outlaw           |         2006 |    149 |            29.99
 Right Cranes           |         2006 |    153 |            29.99
 Quest Mussolini        |         2006 |    177 |            29.99
 Poseidon Forever       |         2006 |    159 |            29.99
 Loathing Legally       |         2006 |    140 |            29.99
 Lawless Vision         |         2006 |    181 |            29.99
 Jingle Sagebrush       |         2006 |    124 |            29.99
 Jericho Mulan          |         2006 |    171 |            29.99
 Japanese Run           |         2006 |    135 |            29.99
 Gilmore Boiled         |         2006 |    163 |            29.99
 Floats Garden          |         2006 |    145 |            29.99
 Fantasia Park          |         2006 |    131 |            29.99
 Extraordinary Conquerer |        2006 |    122 |            29.99
 Everyone Craft         |         2006 |    163 |            29.99
 Dirty Ace              |         2006 |    147 |            29.99
 Clyde Theory           |         2006 |    139 |            29.99
 Clockwork Paradise     |         2006 |    143 |            29.99
 Ballroom Mockingbird   |         2006 |    173 |            29.99
(25 rows)
```

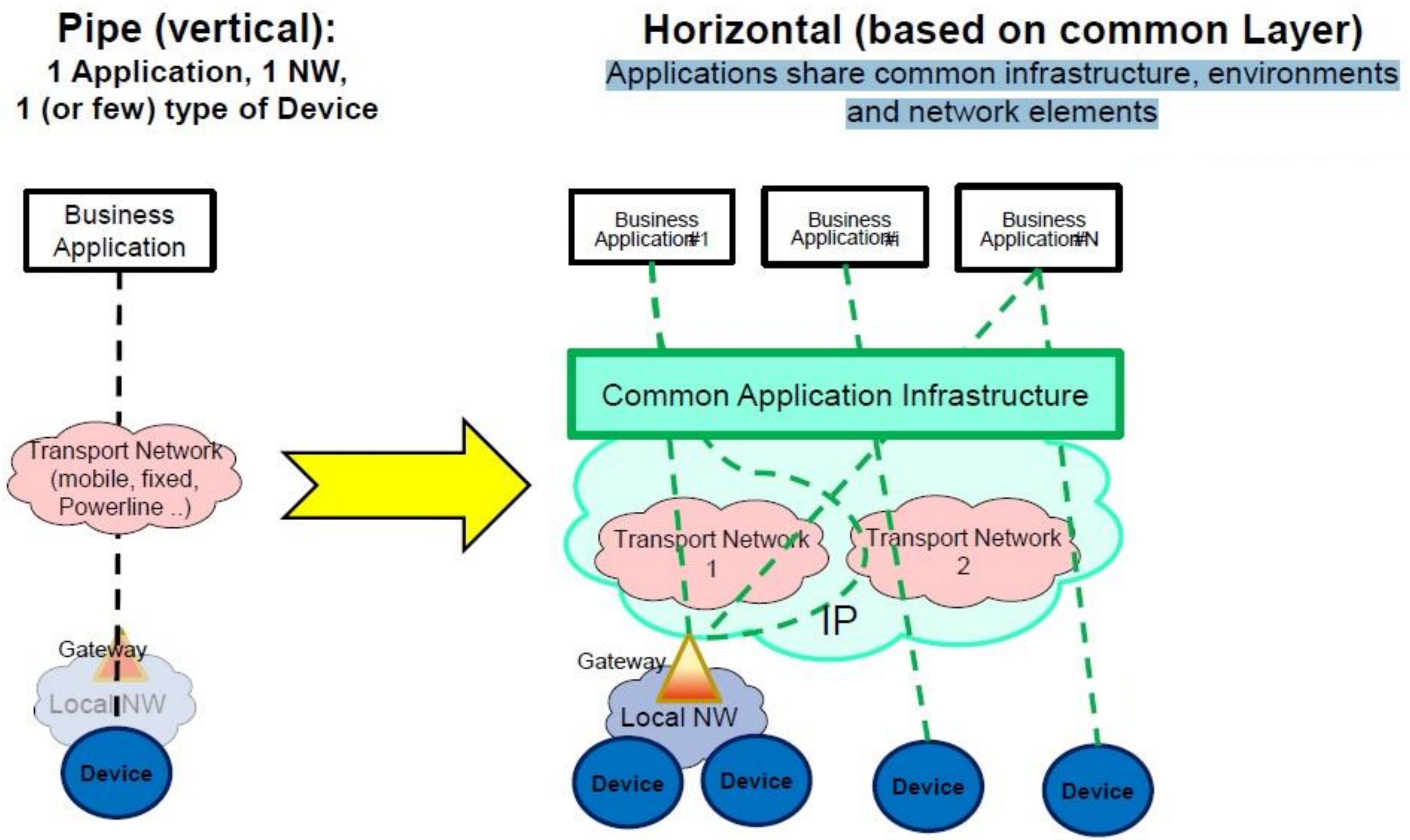
Examples: Oracle, MySQL, Microsoft SQL Server, PostgreSQL, IBM DB2, SQLite

NoSQL and NewSQL Databases

- NoSQL databases – massively distributed, horizontally scalable, fast, do not require fixed table schemas, avoid join operations by storing denormalized data.
- CAP theorem: it is impossible for a distributed system to simultaneously provide consistency, availability, and partition tolerance guarantees –> eventual consistency = high availability and partition tolerance with a reduced level of data consistency.
- NewSQL is a class of modern relational databases that aims to provide the same scalable performance of NoSQL systems for online transaction processing (read-write) workloads while still using SQL and maintaining the ACID guarantees of a traditional database system.

Source: Wikipedia - https://en.wikipedia.org/wiki/Database

Vertical vs. Horizontal Scaling



http://www.iot-week.eu/overview_2013/iot-week-2012/programme-1/tuesday-1/presentations/semantic-interoperability/K1_ETSI-M2M-oneM2M-and-the-need-for-semantics-IoTWeek2012.pdf, By Joerg Swetina (NEC)- ETSI M2M / oneM2M and the need for semantics

NoSQL and NewSQL Database Examples

Type	Notable examples of this type
Wide column	Accumulo, Cassandra, Scylla, HBase
Document:	Apache CouchDB, ArangoDB, BaseX, Clusterpoint, Couchbase, Cosmos DB, eXist-db, IBM Domino, MarkLogic, MongoDB, OrientDB, Qizx, RethinkDB
Key–value:	Aerospike, Apache Ignite, ArangoDB, Berkeley DB, Couchbase, Dynamo, FoundationDB, InfinityDB, MemcacheDB, MUMPS, Oracle NoSQL Database, OrientDB, Redis, Riak, SciDB, SDBM/Flat File dbm, ZooKeeper
Graph:	AllegroGraph, ArangoDB, InfiniteGraph, Apache Giraph, MarkLogic, Neo4J, OrientDB, Virtuoso
New SQL	CockroachDB, Citus, Vitess

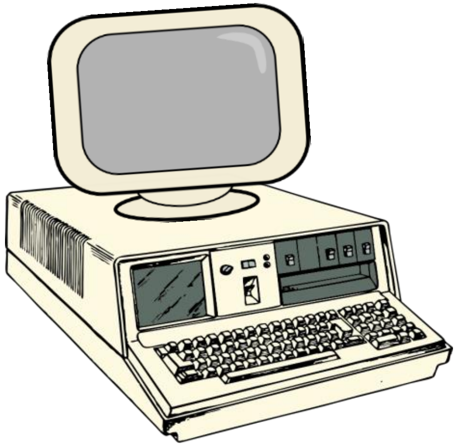
Source: Wikipedia - https://en.wikipedia.org/wiki/NoSQL

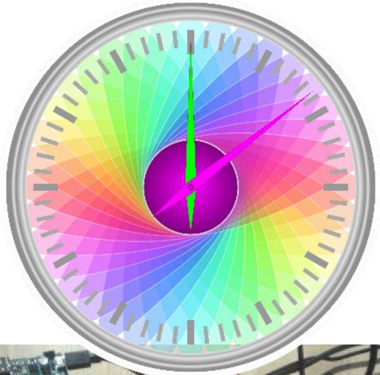
SQL and NoSQL Databases Comparison

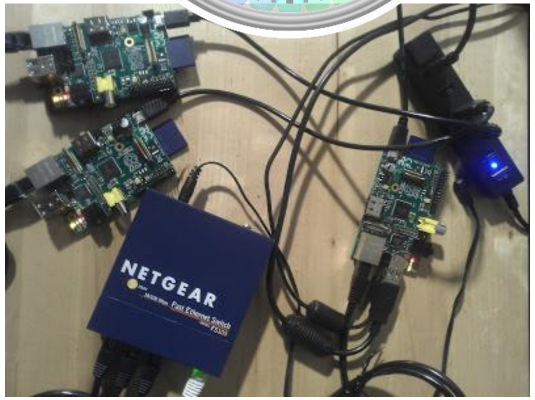
Data model	Performance	Scalability	Flexibility	Complexity	Functionality
Key–value store	high	high	high	none	variable (none)
Column-oriented store	high	high	moderate	low	minimal
Document-oriented store	high	variable (high)	high	low	variable (low)
Graph database	variable	variable	high	high	graph theory
Relational database	variable	variable	low	moderate	relational algebra

Source: Wikipedia - https://en.wikipedia.org/wiki/NoSQL

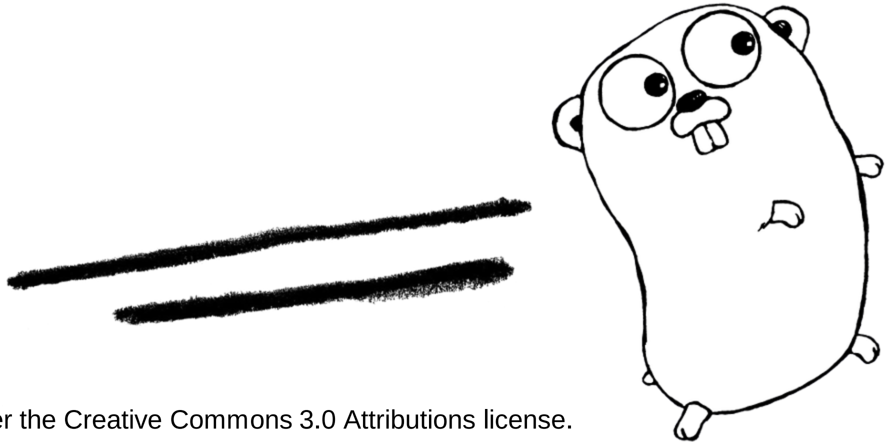
Need for Speed :)







?



Batch Processing

Extract Transform Load

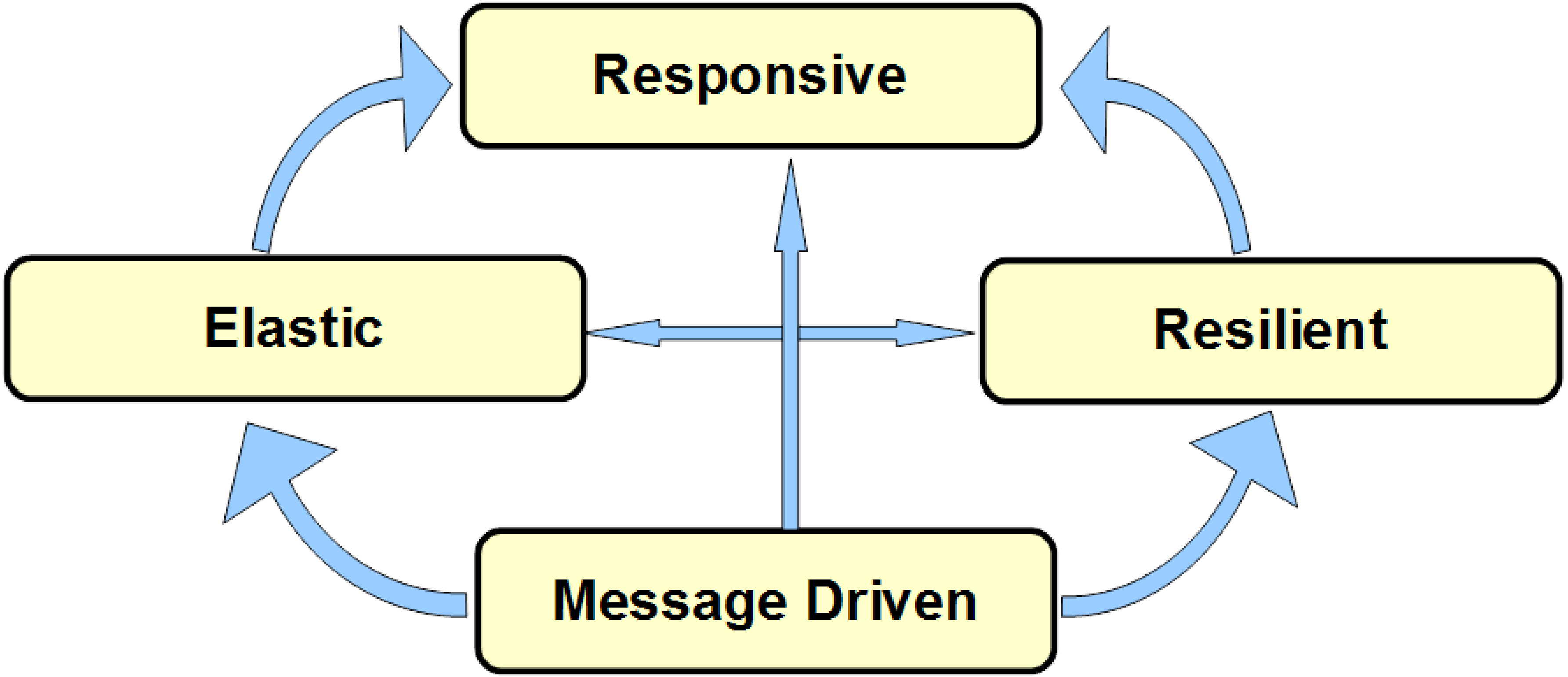
Extract Load Transform Transform Transform

Key Differences between ETL and ELT
[https://www.guru99.com/etl-vs-elt.html]

- ETL stands for Extract, Transform and Load while ELT stands for Extract, Load, Transform.
- ETL loads data first into the staging server and then into the target system whereas ELT loads data directly into the target system.
- ETL model is used for on-premises, relational and structured data while ELT is used for scalable cloud structured and unstructured data sources.
- Comparing ELT vs. ETL, ETL is mainly used for a small amount of data whereas ELT is used for large amounts of data.
- When we compare ETL versus ELT, ETL doesn’t provide data lake supports while ELT provides data lake support.
- Comparing ELT vs ETL, ETL is easy to implement whereas ELT requires niche skills to implement and maintain.

Reactive Manifesto

[http://www.reactivemanifesto.org]



Scalable, Massively Concurrent

- **Message Driven** – asynchronous message-passing allows to establish a boundary between components that ensures loose coupling, isolation, location transparency, and provides the means to delegate errors as messages [Reactive Manifesto].
- The main idea is to separate concurrent producer and consumer workers by using **message queues.**
- **Message queues** can be **unbounded** or **bounded** (limited max number of messages)
- **Unbounded** message queues can present memory allocation problem in case the producers outrun the consumers for a long period → **OutOfMemoryError**

Data / Event / Message Streams

"Conceptually, a stream is a (potentially never-ending) flow of data records, and a transformation is an operation that takes one or more streams as input, and produces one or more output streams as a result."

Apache Flink: Dataflow Programming Model

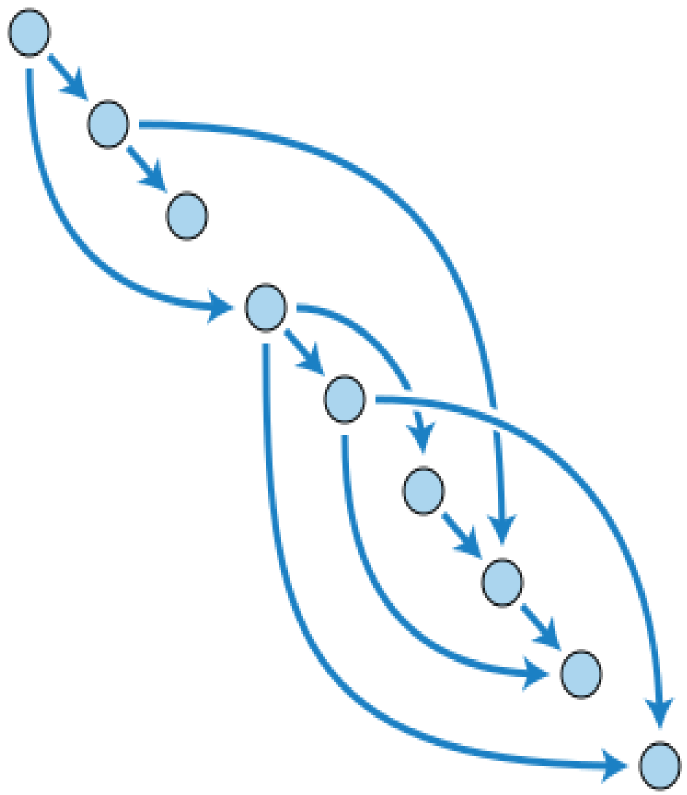
Data Stream Programming

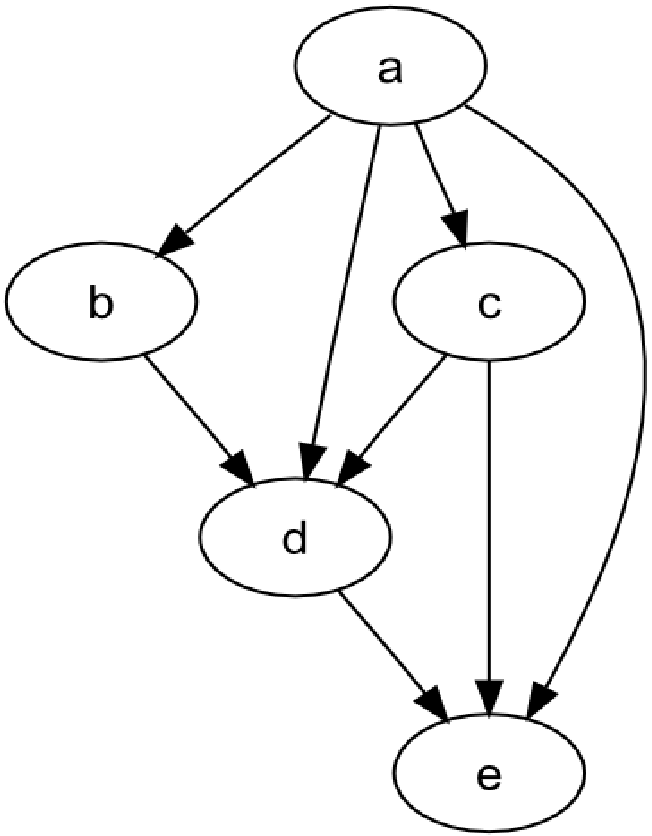
The idea of abstracting logic from execution is hardly new -- it was the dream of SOA. And the recent emergence of microservices and containers shows that the dream still lives on.

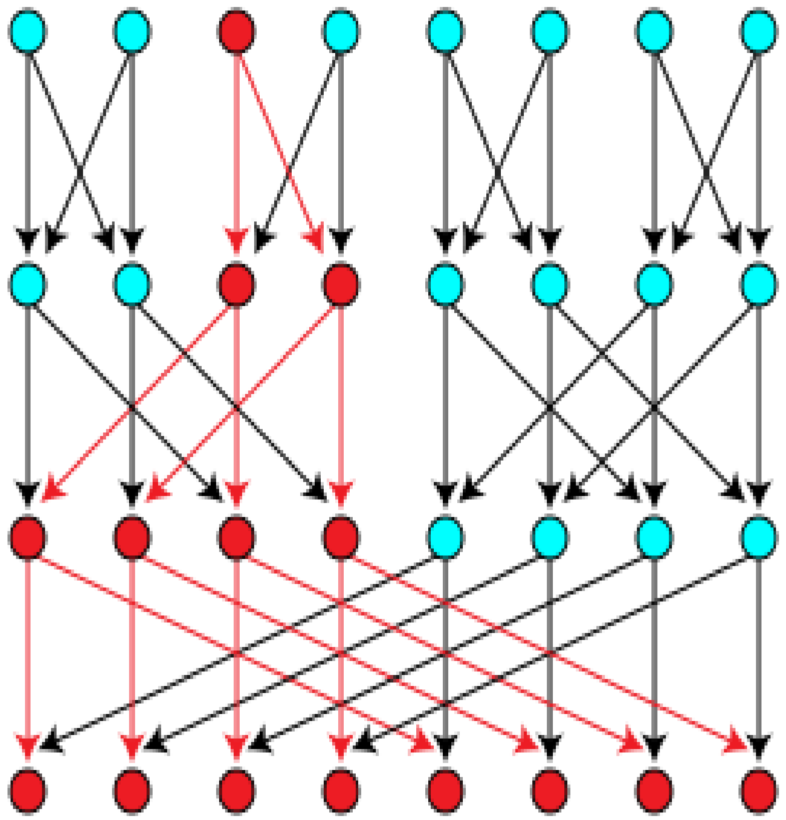
For developers, the question is whether they want to learn yet one more layer of abstraction to their coding. On one hand, there's the elusive promise of a common API to streaming engines that in theory should let you mix and match, or swap in and swap out.

Tony Baer (Ovum) @ ZDNet - Apache Beam and Spark: New coopetition for squashing the Lambda Architecture?

Direct Acyclic Graphs - DAG

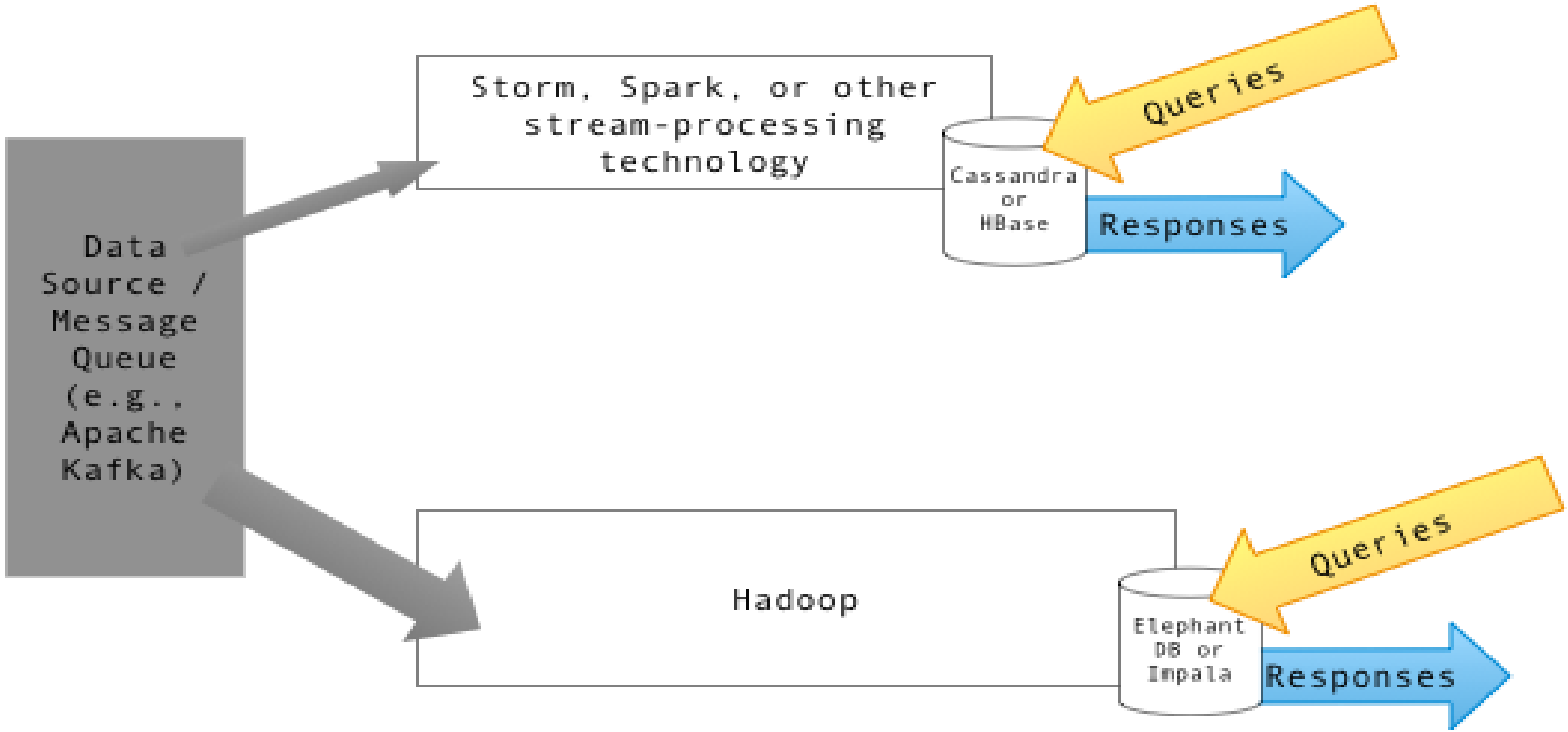






Lambda Architecture - I

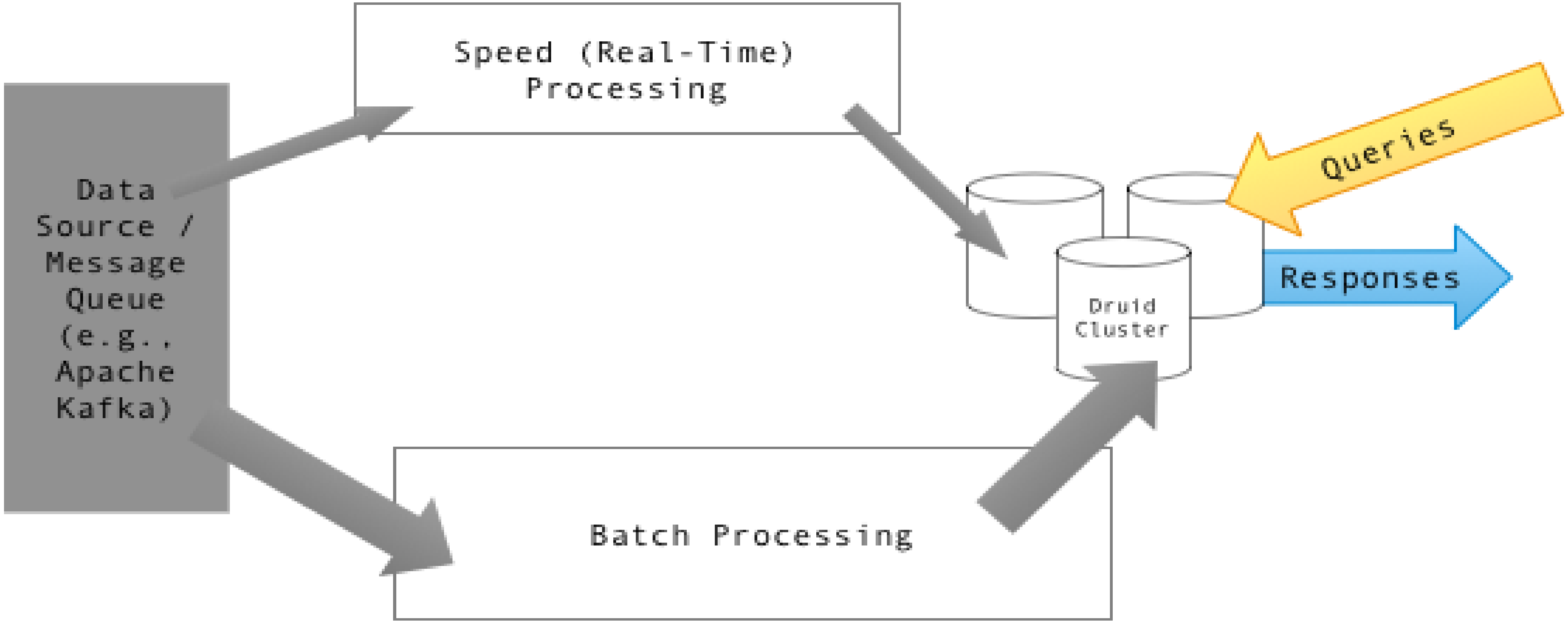
Query = λ (Complete data) = λ (live streaming data) * λ (Stored data)



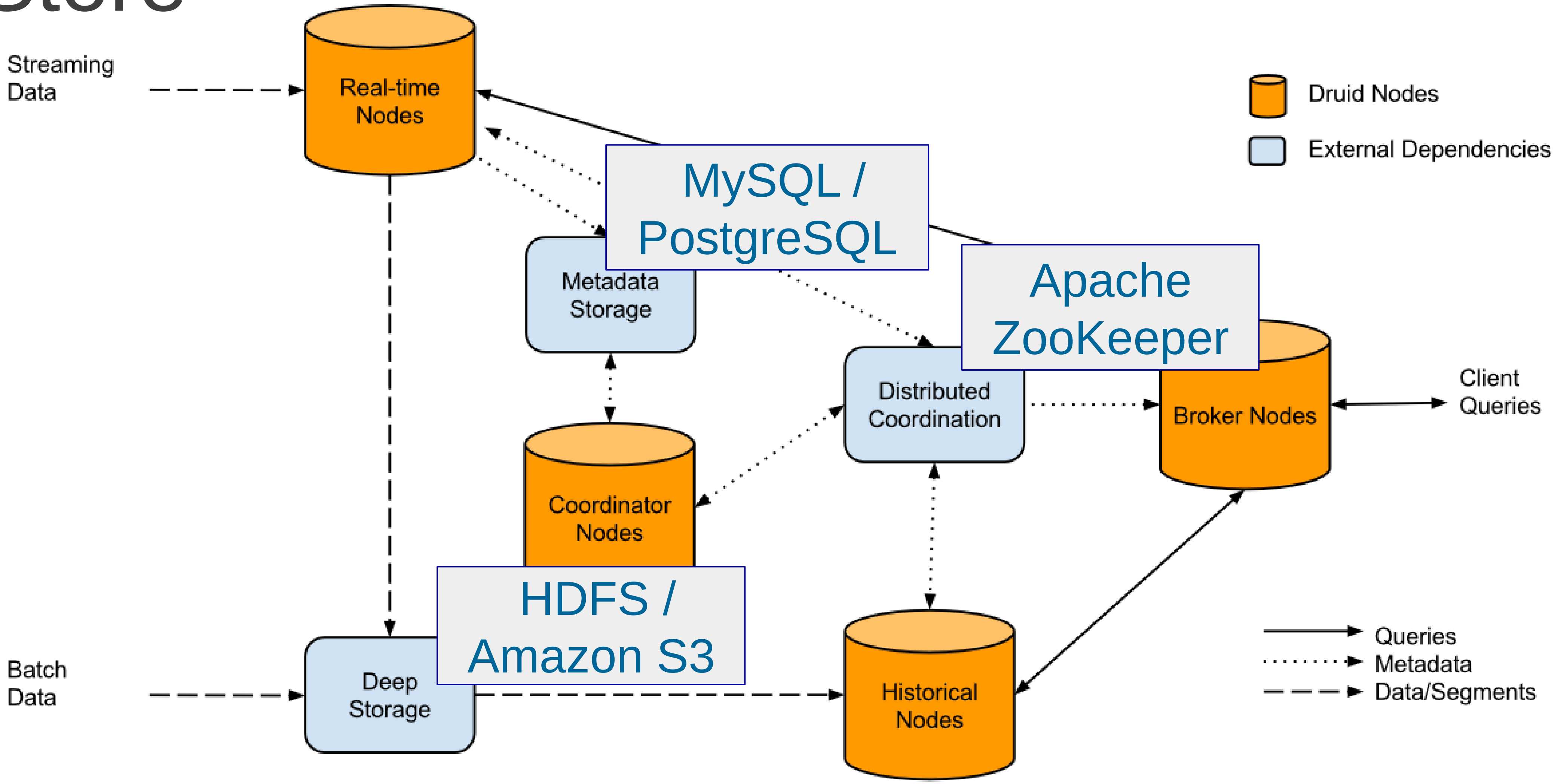
https://commons.wikimedia.org/w/index.php?curid=34963987, By Textractor - Own work, CC BY-SA 4

Lambda Architecture - II

Query = λ (Complete data) = λ (live streaming data) * λ (Stored data)



Lambda Architecture - Druid Distributed Data Store



https://commons.wikimedia.org/w/index.php?curid=33899448 By Fangjin Yang, GFDL

Kappa Architecture

Query = K (New Data) = K (Live streaming data)

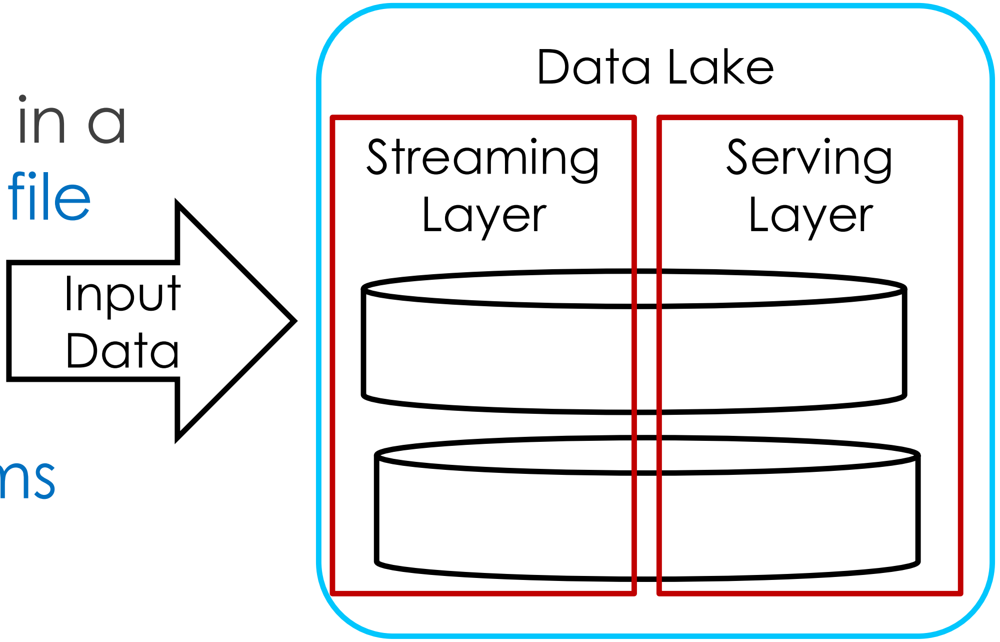
λ **vs** $\varkappa$

- Proposed by Jay Kreps in 2014
- Real-time processing of distinct events
- Drawbacks of Lambda architecture:
  - It can result in coding overhead due to comprehensive processing
  - Re-processes every batch cycle which may not be always beneficial
  - Lambda architecture modeled data can be difficult to migrate
- Canonical data store in a Kappa Architecture system is an append-only immutable log (like Kafka, Pulsar)

Kappa Architecture II

Query = K (New Data) = K (Live streaming data)

- Multiple data events or queries are logged in a queue to be catered against a distributed file system storage or history.
- The order of the events and queries is not predetermined. Stream processing platforms can interact with database at any time.
- It is resilient and highly available as handling terabytes of storage is required for each node of the system to support replication.
- Machine learning is done on the real time basis



Distributed Stream Processing – Apache Projects:

- Apache Spark is an open-source cluster-computing framework. Spark Streaming, Spark Mllib
- Apache Storm is a distributed stream processing – streams DAG
- Apache Samza is a distributed real-time stream processing framework.

Distributed Stream Processing – Apache Projects II

- Apache Flink - open source stream processing framework – Java, Scala
- Apache Kafka - open-source stream processing (Kafka Streams), real-time, low-latency, high-throughput, massively scalable pub/sub
- Apache Beam – unified batch and streaming, portable, extensible

Zeta Architecture

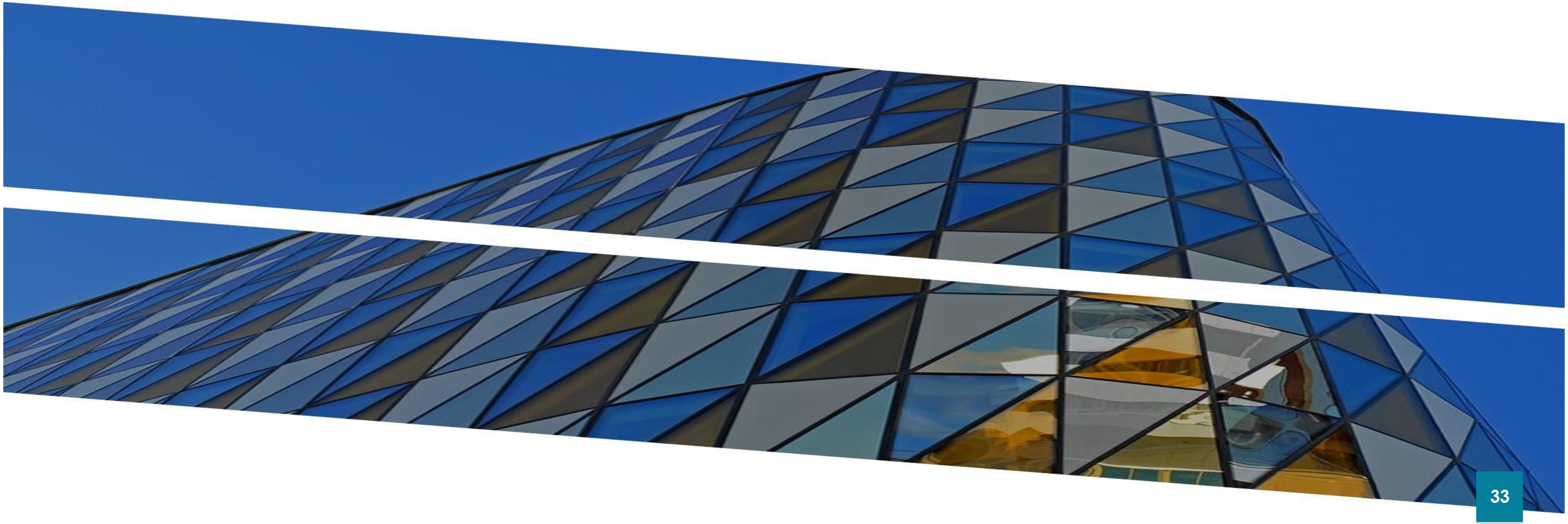
Zeta Architecture

- Main characteristics of Zeta architecture:
  - file system (HDFS, S3, GoogleFS),
  - realtime data storage (HBase, Spanner, BigTable),
  - modular processing model and platform (MapReduce, Spark, Drill, BigQuery),
  - containerization and deployment (cgroups, Docker, Kubernetes),
  - Software solution architecture (serverless computing),
- Recommender systems and machine learning
- Business applications and dynamic global resource management (Mesos + Myriad, YARN, Diego, Borg).
- Labs – introduction to Docker, Docker-Compose and object storage with MINIO.

Data Lakes

- A data lake is a system or repository of data stored in its natural/raw format, usually object blobs or files.
- A data lake is usually a single store of data including raw copies of source system data, sensor data, social data etc., and transformed data used for tasks such as reporting, visualization, advanced analytics and machine learning.
- A data lake can include structured data from relational databases (rows and columns), semi-structured data (CSV, logs, XML, JSON), unstructured data (emails, documents, PDFs) and binary data (images, audio, video).
- A data lake can be established "on premises" (within an organization's data centers) or "in the cloud" (using cloud services from vendors such as Amazon, Microsoft, or Google).

Big Data Storage Solutions



Categores of solutions for big data storage

- Block
    - Where everything is in fixed-size chunks
    - SCSI and SCSI-based protocols, and how FC and iSCSI fit in
- Files
    - When everything is a stream of bytes
    - NFS and SMB
- Objects
    - When everything is a blob
    - HTTP, key value and RESTful interfaces, Amazon S3
- Altogether
    - When files, blocks and objects collide
    - A data swamp is a deteriorated and unmanaged data lake that is either inaccessible to its intended users or is providing little value.

Block-level storage

- Block-level storage is a concept in cloud-hosted data persistence where cloud services emulate the behaviour of a traditional block device, such as a physical hard drive. It is a form of network-attached storage (NAS).
- Storage in such services is organised as blocks. This emulates the type of behaviour seen in traditional disks or tape storage through storage virtualization. Blocks are identified by an arbitrary and assigned identifier by which they may be stored and retrieved, but this has no obvious meaning in terms of files or documents. A filesystem must be applied on top of the block-level storage to map 'files' onto a sequence of blocks.
- Amazon EBS (Elastic Block Store) is an example of a cloud block store.[3] Cloud block-level storage will usually offer facilities such as replication for reliability, or backup services.

Block-level storage

- Block-level storage is in contrast to an object store or 'bucket store', such as Amazon S3 (Simple Storage Service), or to a database. These operate at a higher level of abstraction and are able to work with entities such as files, documents, images, videos or database records.[5]
- Instance stores are another form of cloud-hosted block-level storage. These are provided as part of an 'instance', such as an Amazon EC2 (Elastic Compute Cloud) service.[6] As EC2 instances are primarily provided as compute resources, rather than storage resources, their storage is less robust. Their contents will be lost if the cloud instance is stopped.
- At one time, block-level storage was provided by storage area networks (SAN) and NAS provided file-level storage. With the shift from on-premises hosting to cloud services, this distinction has shifted. Even block-storage is now seen as distinct servers (thus NAS), rather than the previous array of bare discs.

File Storage [https://www.redhat.com/en/topics/data-storage/file-block-object-storage, https://www.ibm.com/cloud/blog/object-vs-file-vs-block-storage]

- File storage, also called file-level or file-based storage, is exactly what you think it might be: Data is stored as a single piece of information inside a folder, just like you'd organize pieces of paper inside a manila folder. When you need to access that piece of data, your computer needs to know the path to find it. Data stored in files is organized and retrieved using a limited amount of metadata that tells the computer exactly where the file itself is kept. It's like a library card catalog
- File storage is when all the data is saved together in a single file with a file extension type that's determined by the application used to create the file or file type, such as .jpg, .docx or .txt. For example, when you save a document on a corporate network or your computer's hard drive, you are using file storage. Files may also be stored on a network-attached storage (NAS) device. These devices are specific to file storage, making it a faster option than general network servers. Other examples of file storage devices include cloud-based file storage systems, network drives, computer hard drives and flash drives.

Object Storage – Amazon S3, MINIO

- Object storage (also known as object-based storage[1]) is a computer data storage architecture that manages data as objects, as opposed to other storage architectures like file systems which manages data as a file hierarchy, and block storage which manages data as blocks within sectors and tracks.[2] Each object typically includes the data itself, a variable amount of metadata, and a globally unique identifier.
- Object storage can be implemented at multiple levels, including the device level (object-storage device), the system level, and the interface level. In each case, object storage seeks to enable capabilities not addressed by other storage architectures, like interfaces that are directly programmable by the application, a namespace that can span multiple instances of physical hardware, and data-management functions like data replication and data distribution at object-level granularity.
- Object storage systems allow retention of massive amounts of unstructured data. Object storage is used for purposes such as storing photos on Facebook, songs on Spotify, or files in online collaboration services, such as Dropbox.[3]

Thank’s for Your Attention!

Trayan Iliev

IPT – Intellectual Products & Technologies

http://iproduct.org/

https://github.com/iproduct

https://twitter.com/trayaniliev

https://www.facebook.com/IPT.EACAD