CLOUDERA

Future Data Lake Architecture

Cloud ready, robust, governed, containerised, scalable

Johannes Muselaers
Solutions Engineer Nordics
jmuselaers@cloudera.com
+46 72 588 1091

Agenda

Follow up meeting

- Introductions
- Cloudera news & Marketplace update
- New EDW
- Teradata offload
- Cloudera License Changes
- Cloudera Product Roadmap
- Next meeting

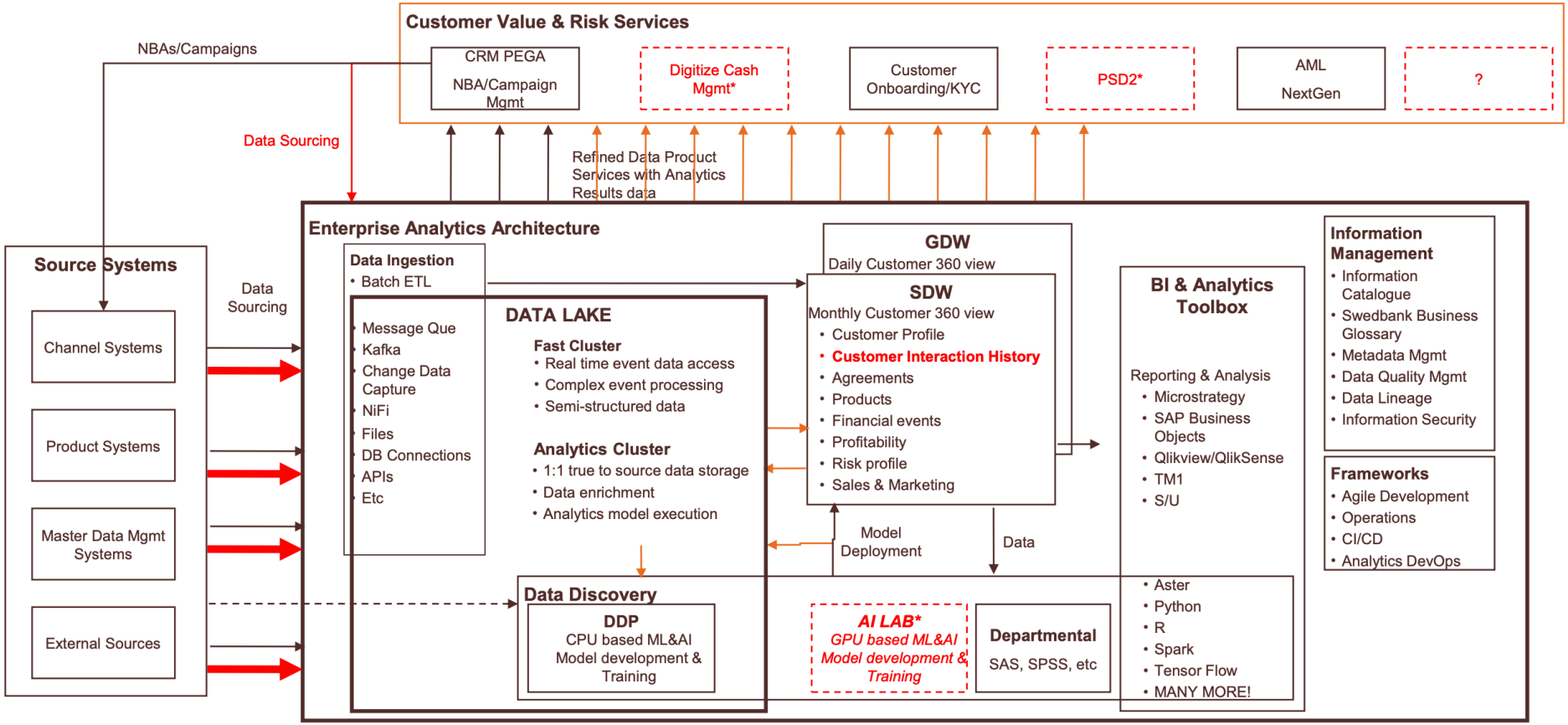
PDF of this presentation:

https://github.com/jmuselaers/Presentations/FollowUpMeeting.pdf

NEW COMPANY LOGO

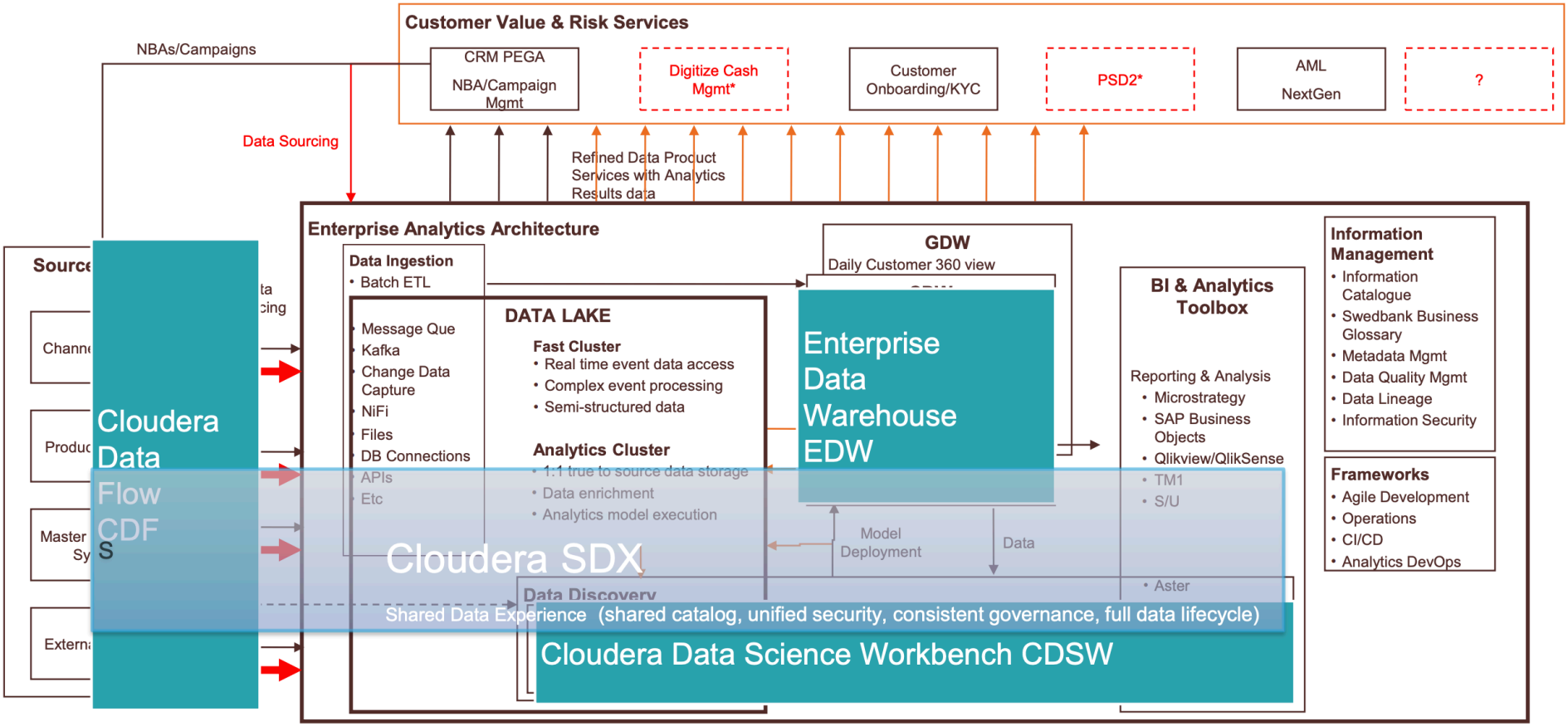
Enterprise Analytics Architecture 2018-2020

* Not decided yet/potential use cases



Enterprise Analytics Architecture 2018-2020

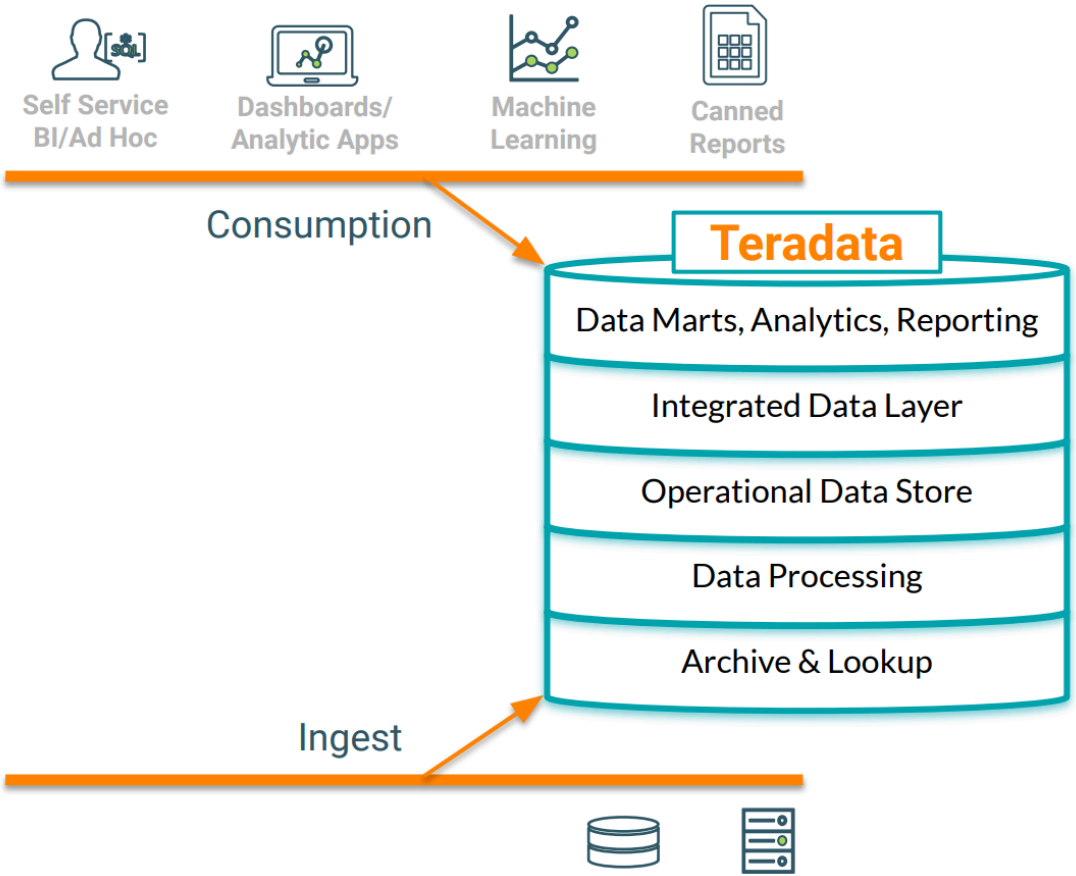
* Not decided yet/potential use cases



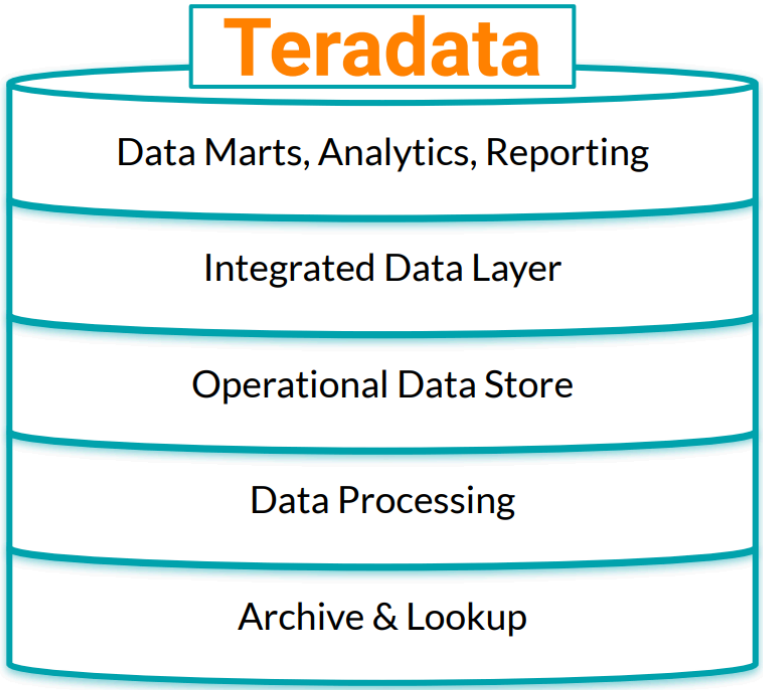
Teradata Offload

Staged approach

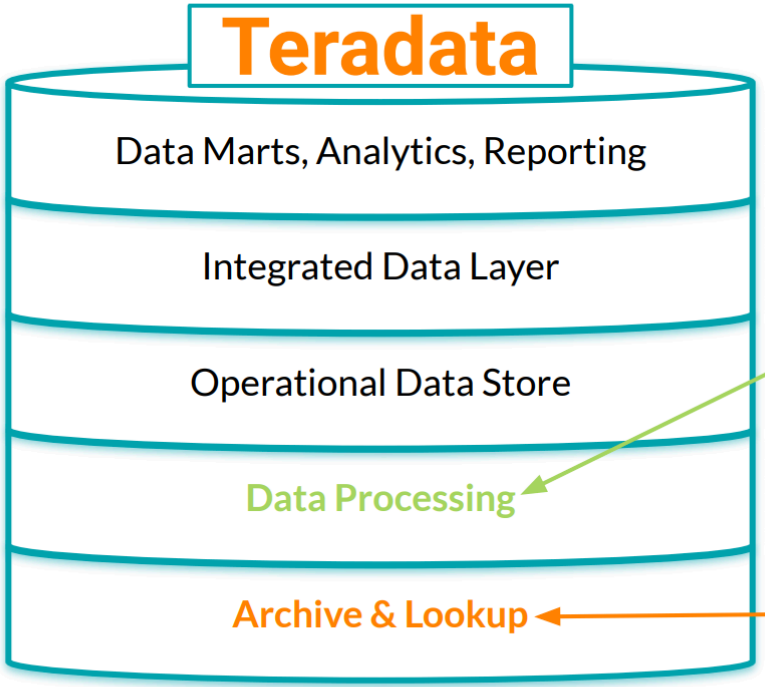
CURRENT ARCHITECTURE



MOST COMMON WORKLOAD TYPES IN A TERADATA ENVIRONMENT



VALUE OF MIGRATING TERADATA WORKLOADS TO CLOUDERA



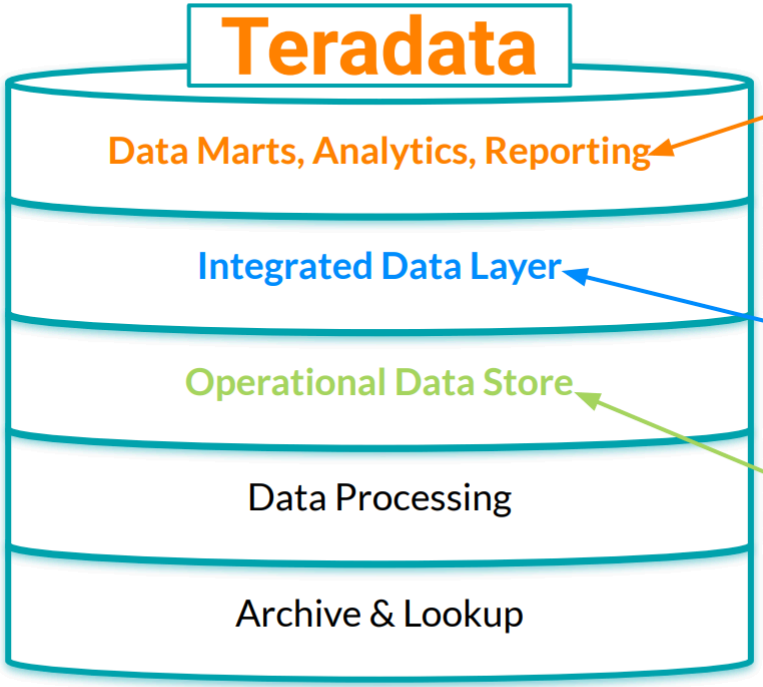
DATA PROCESSING

- Much Faster Processing for Quicker Time-to-Query
- More Scalable - Leverage Ever-Growing Volumes
- Use Your Current ETL Tools or Re-purpose Your Custom Code
- Common Starting Point - Good ROI

ARCHIVE & LOOKUP

- Cost-per-TB up to 10x cheaper for long term storage
- Active archive option to get additional analytic value
- Common starting point; good ROI

VALUE OF MIGRATING TERADATA WORKLOADS TO CLOUDERA



DATA MARTS, ANALYTICS, REPORTING

- Better Performance at Scale
- More Flexible Self-service Querying
- Workload Introspection - Tuning, Troubleshooting, Isolation
- Dramatically Expand Use Cases, Users

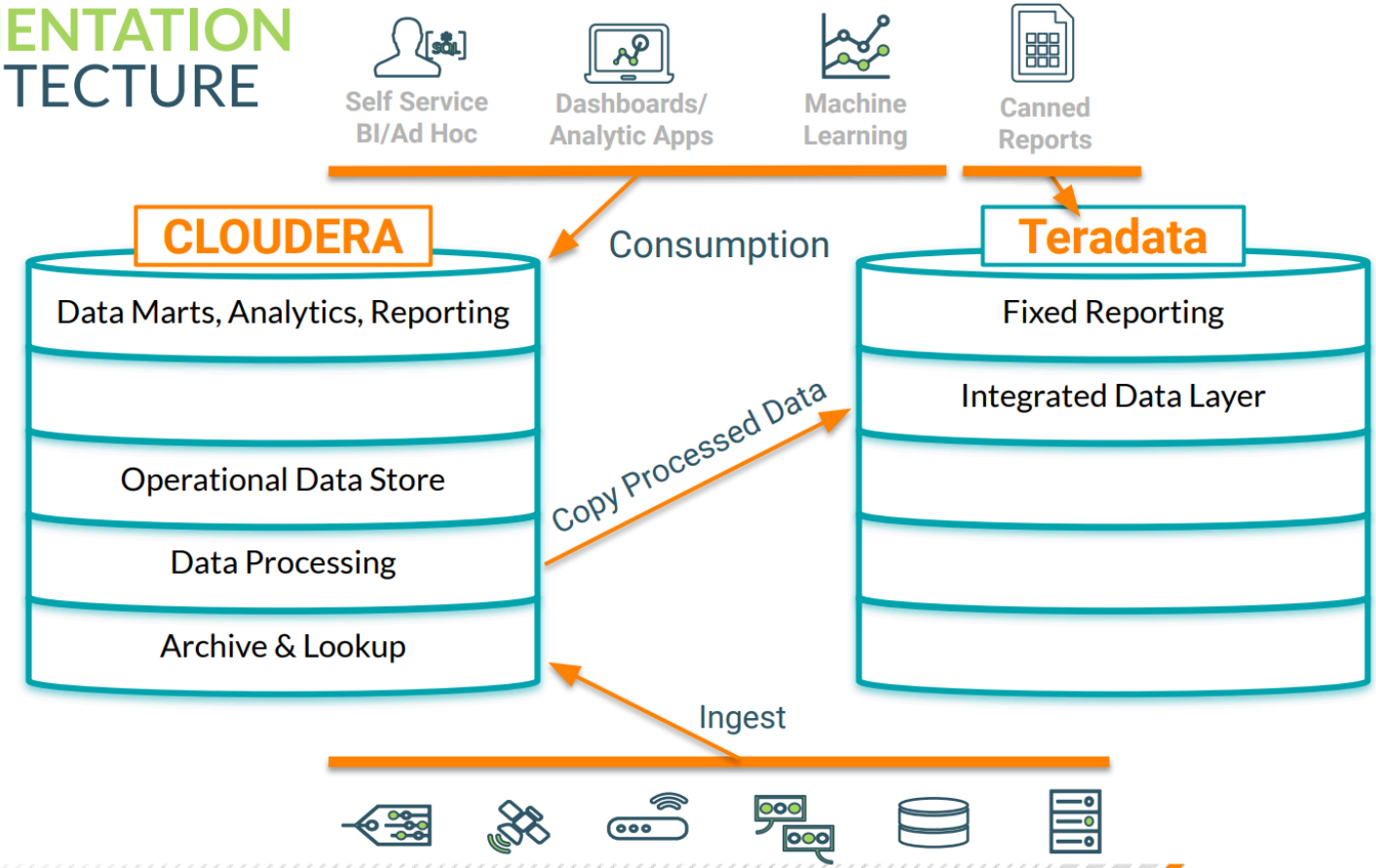
INTEGRATED DATA LAYER

- Go Beyond Relational - Store More Data
- Model Data for Future Self-service Use
- Opportunity for Data Transformation

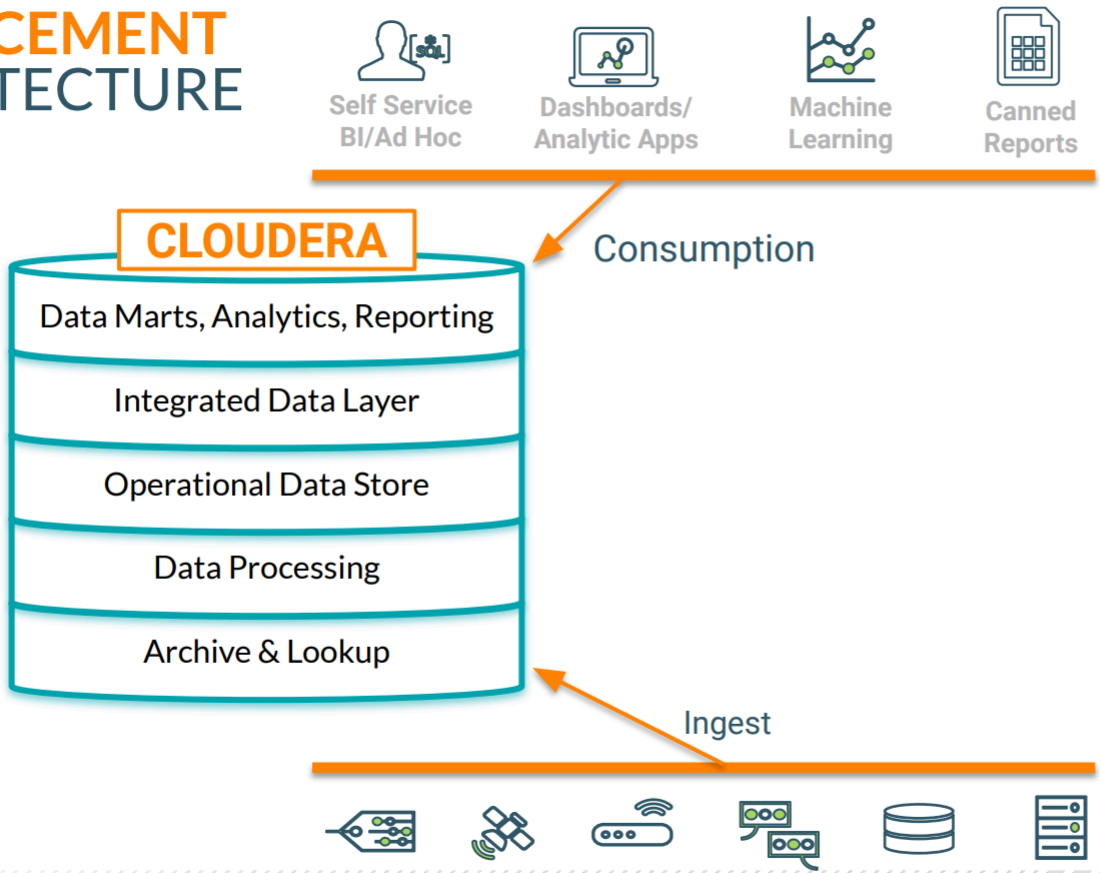
OPERATIONAL DATA STORE

- Use deeper history in ODS reporting
- Incorporate Less-structured with Transactional

AUGMENTATION ARCHITECTURE



REPLACEMENT ARCHITECTURE



Backup slides

Key requirements and descriptions √

1. Self-service and UI driven workflow for data scientists (1,2,1)

- We need freedom for Data Scientist to easily administrate their own work without the need to have an administrator do it. We want the developers to self-service as much as possible to be more effective. This includes having an easy scheduler that uses the best parallelization framework (CollectiveAllReduce) without Data Scientists need to write code for it. We want the data scientists to focus on developing models and not scheduling training against specific GPUs or figuring out how to distribute the load most effectively.

2. Build a model (5,3,1)

- We want to support end to end deep learning. We should handle parallel experiments for hyper parameter tuning, distributed training, monitoring training using tensor board together with model serving for testing and systems outside of ODL. The ability to rapidly test new big data technologies against new and old models to make them better and more effective is very important to decrease TTM. Monitoring during model training is needed to shorten the time for feedback and redevelopment.
- Data pre-processing will be done within the DDP platform and final feature generation and iterations will be done in AI Lab.

3. Multi developer, multi-tenant separation and collaboration (4,2,4)

- Data needs to be isolated between developers so that each developer only has access to its own data. Project based isolation so that developers from different tenants can work together during a project and share data with each other. It should be easy to work together in a secure environment. Development should be containerized, so it is possible for projects to run their own version of Tensorflow, Python or other frameworks.

4. Horizontal scalable HW platform (3,4,5)

We should be able as the demand requires add more HW without the need to change the SW architecture. There should be little to no change for the Data Scientists working only the capacity within the platform has grown.

5. Data acquisition from DDP (1,7,7)

Capacity to schedule a data pipeline from another cluster, which should be user/system based. For the POC we will do a one-time copy of data needed for the deep learning use cases to make them successful, there is client in place that can perform this task.

5. Feature Store (6, 6,3)

- The Feature Store is a central vault for documented, curated, and access-controlled features. The Feature Store solves the problem of ad-hoc and siloed machine learning pipelines, where features, the training data for such pipelines, tend to become disorganized, disjointed, and duplicated, leading to correctness problems and redundant work. The Feature Store also gives Enterprises full Machine Learning
- Governance - the exercise of authority and control (access, monitoring, auditing, and provenance) over the management of machine learning assets. Repeatable experiments, features, and models can all be governed and managed by Hopsworks.

7. Deployment schema (5,6,6)

- The analytical model developed by Data Scientists will follow the life-cycle of the Analytical DevOps process. There will be Test/Development and Production environments where all models needs to go through.

HDP VS. CDP DATA CENTER (1 of 2)

Component	HDP 2.6.5	HDP 3.1.4	Runtime 7.0
Apache Accumulo	1.7.0	1.7.0	[Roadmap]
Apache Atlas	0.8.0	1.1.0	2.0.0
Apache Flume	1.5.2	[Replaced with NiFi]	[Replaced with NiFi]
Apache Hadoop	2.7.3	3.1.1	3.1
Apache HBase	1.1.2	2.0.2	2.2
Apache Hive	1.2.1 / 2.1.0	3.1.0	3.1
Apache Knox	0.12	1.0.0	1.3
Apache Livy	-	0.5.0	0.5
Apache Oozie	4.2.0	4.3.1	5.1
Apache Phoenix	4.7.0	5.0.0	[Roadmap]

HDP VS. CDP DATA CENTER (2 of 2)

Component	HDP 2.6.5	HDP 3.1.4	Runtime 7.0
Apache Pig	0.16	0.16	[Roadmap]
Apache Ranger	0.7.0	1.2.0	1.2.0
Apache Spark	1.6.3 / 2.3.2	2.4	2.4
Apache Sqoop	1.4.6	1.4.7	1.4.7
Apache Storm	1.1.0	1.2.1	[Replaced with Spark Streaming]
Apache TEZ	0.7.0	0.9.1	0.9
Apache Zeppelin	0.7.3	0.8.0	0.8
Apache ZooKeeper	3.4.6	3.4.6	3.4.6

THANK YOU

CLOUDERA