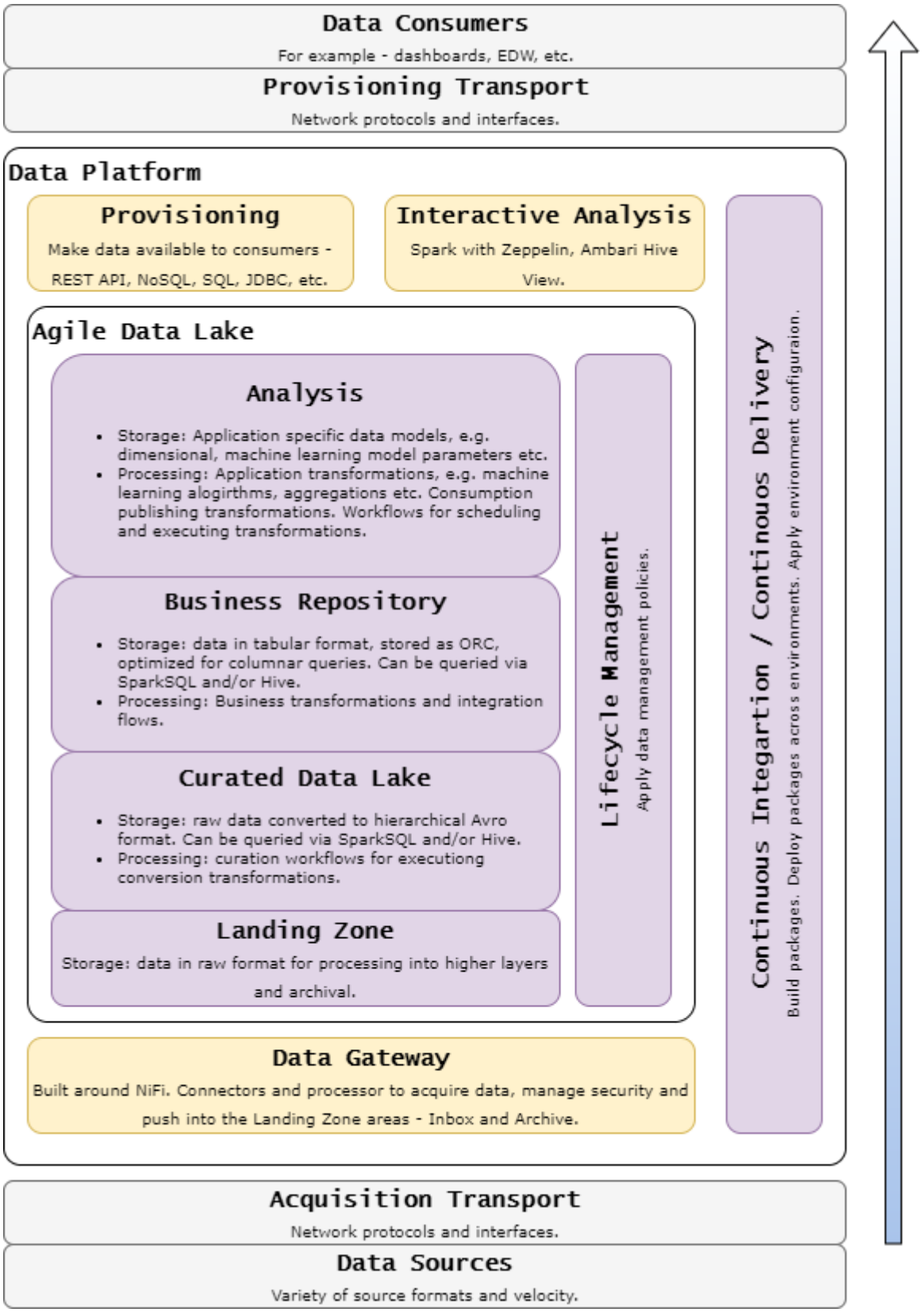
SDP Architecture Overview

The architecture of the platform follows layered structure. Following diagram presents the platform from developer and business user perspective:



The platform follows layered architecture. The data flows between the layers, entering the platform through the Data Gateway. The higher is the layer, the more insight is stored with the data.

Data Gateway

The Data Gateway is used to connect to remote systems, using various transport protocols, acquire data from the sources and store it in the Landing Zone. It is a separate tier. The whole process is referred as data ingestion. Central component in the Data Gateway is HDF and NiFi in particular. Data sets are ingested from source systems via data ingestion flows.

NiFi is a flow management platform for collecting and transporting data from sources into the SDP. NiFi provides the data acquisition, simple data management, transport and delivery mechanism designed to accommodate the diverse dataflows generated by a world of connected people, systems, and things.

Data ingestion flows acquire data from the remote systems. If source data is compressed or packaged in archives, it is extracted into distinct flow files. Further security management policies are applied, e.g. anonymization. Managed data is pushed into the Inbox area of the Landing Zone for retrieval into the Curated Data Lake. To enable recovery managed data is packaged and compressed and stored into the Archive area of the Landing Zone.

Data ingestion flows are discussed in more details in the following chapters.

Agile Data Lake

Agile Data Lake (also referred simply as Data Lake) provides distributed, highly available storage and distributed data processing capabilities. The platform is a hybrid - built with on-premise and on-cloud services. The services are based on the Hadoop ecosystem. Although the tools emerge from Hadoop and HDFS is used for storage, the platform architecture is not tightly coupled with Hadoop. The Data Lake provides:

- Scalable and reliable data storage. Implemented on top of Hadoop File System (HDFS), Azure Blob Storage or Azure Data Lake Storage (ADLS).
- Parallel processing capabilities with horizontal scaling. Provided by Spark.
- Database-like metadata catalog, using HCatalog.
- Low-latency, SQL-like query capabilities with JDBC interface. Available through Hive.
- Ad-hoc analytical capabilities, exposed through Zeppelin with Spark (and Livy) interpreter.
- Resource/task management and scheduling handled by YARN.
- Workflow management and scheduling handled by Oozie.

From developer and business user perspective, Agile Data Lake is composed by a series of layers. Agile Data Lake layers are discussed in details in following sections.

Landing Zone

Landing Zone is a storage layer with two areas - Inbox and Archive. The layer is passive. Provides isolation from the source systems.

Inbox area is a short term storage for ingested data to be picked up, further processed and consumed by the curation workflows. Inbox is not a subject of lifecycle management.

Archive area is stores archived and compressed data for longer periods. It allows for recovery in case of system malfunction. Typically Archive area is at alternate, independent (usually physically remote) location.

Curated Data Lake

Curated Data Lake provides storage and processing capabilities. The storage holds source data converted to Avro, JSON schema definitions. Data is registered in HCatalog and can be queried using SQL (with LATERAL VIEW extension for querying nested and complex structures). Metadata from HCatalog can be used for executing queries using Hive or SparkSQL.

Processing capabilities include platform converters from XML, JSON and CSV to Avro. Converters are capable to infer the raw data files schema, but in some complex situations, the inference doesn't work. In these cases the schema is provided as configuration to the converters. The schema is in JSON format. The same schema can be used to create external Hive tables at the Curated Data Lake and enable running queries on the source data.

Curation Flows convert data at the Landing Zone from raw format to Avro format. The process of this transformation is referred as curation. Curation process needs converters to Avro, Oozie workflows to schedule and execute the conversion and schema files as configuration for the conversion process.

Business Repository

Business Repository provides storage and processing capabilities. Storage keeps cleansed, conformed data, with metadata registered in HCatalog. Big parts of the Business Repository data has enterprise value and is shared between multiple applications. Data is stored as flat tables in ORC format which allows for efficient columnar queries. In some cases other formats might be justified, e.g. Parquet or Avro.

Processing capabilities include business transformations for transforming data from Curated Data Lake to the Business Repository. For straightforward business transformations which can be expressed using SQL, platform SQL Transformer is provided. Complex transformations might be implemented with dedicated transformers, using Spark or another programming framework.

Analysis Layer

Analysis Layer provides application specific storage. Each application uses Application Data Models to describe corresponding business domain and are modelled to answer specific questions. Application Datat Models are similar to the traditional Data Marts, but are not limited to any particular modeling methodology, e.g. Dimensional, Snowflake, Data Vault etc.

Transformations that feed this layer are also executed and coordinated by Oozie workflows.

Based on the actual application requirements, the layer also might export data to the Consumption Provisioning layer using publishing transformations. Publishing is executed and coordinated by Oozie workflows.

This layer is closed. This means that higher layers e.g. Consumption Provisioning layer cannot access directly the Business Repository. For pass-through access, Hive views can be used, but the isolation between the layers should not be broken.

Lifecycle Management

The Lifecycle Management layer defines and executes data retention policies across all the layers.

Consumption Provisioning

Different applications might have different provisioning requirements. Based on the application design, the layer can include relational database, NoSQL database, SFTP edge area etc. In the simplest case JDBC interface to the application's Analysis layer is provided.

Continuous Delivery and Continuous Integration

The CI/CD layer includes standard tools: Jenkins, Bitbucket, Ansible, Sonatype Nexus, etc. to perform builds and implement automated continuous delivery and integration pipelines.

Environment Types

The Data Platform is hybrid - a mix of components. From deployment perspective components could be on-premise or on-cloud. From vendor perspective same components could be implemented on different environments, using different (but compatible) vendor software. From software development perspective environments, reflect the software development lifecycle stages - Development (DEV), System Integration Test (SIT), User Acceptance Test (UAT) and Production (PROD).

Agile Data Lake Implementation

HDInsight On-cloud

This implementation is used for processing data on the cloud. HDInsight is provided as PaaS. The storage is not coupled with processing instances which allows for elastic scalability. HDInsight is a cloud version of Hortonworks Data Platform (HDP). This provides very good portability of the processing components, created against HDInsight versus HDP. There is no additional cost associated with HDP licenses for HDInsight.

Storage can be accessed from applications as if it were HDFS as well as using Azure Storage Explorer - web and/or desktop versions. The storage can be accessed directly by NiFi, Azure Data Factory etc. It can also be shared among different clusters.

Storage: Azure Blob or Azure Data Lake Storage (ADLS).

HDP On-premise

This implementation is deployed on dedicated hardware, hosted on-premise. Each machine has attached storage which is used by HDFS. HDFS is available only through the Hadoop's API.

HDP On-cloud

This implementation is used for testing applications, which are required to run on HDP based Agile Data Lake, e.g. HDP On-premise. It uses cloud machine instances (IaaS) with attached dedicated disks which are used for HDFS. HDFS is available only through the Hadoop's API.

Development Environment

DEV environment is controlled by the project development team. It is being used by developers to implement and test various functionalities required to implement the project deliverables. Developers have full access and full control on the DEV environment.

Development environment is HDInsight on-cloud environment. It is created on-demand to serve the needs for a specific project. It is operational during the SLA hours, as defined by the project SLA. Outside the operation hours it is stopped.

System Integration Test Environment

SIT environment is controlled by the project test team. It is being used by the test team to validate builds and developed functionality end-to-end. Access to SIT is restricted:

- Deployment by CI/CD only.
- Execution is triggered on-request by the platform team.
- Development and test team members have read-only access.
- Small data samples are used.

In some projects the tests could be fully automated.

SIT environment is HDInsight on-cloud environment. It is created on-demand for particular project(s) and is operational only during the testing activities. Could be stopped, based on the SLA defined by the project.

User Acceptance Test Environment

UAT environment is controlled by the business team. It is being used by business users to validate application releases. Access to UAT is restricted:

- Deployment by CI/CD only.
- Execution is triggered on-request by the platform team, executed through the scheduler.
- Business users have read-only access. Developers could be granted temporary read-only access.
- Processing is done on a production data, collected over a certain time interval, e.g. a week or a month.

UAT environment is hybrid. Typically:

- On-premise gateway
- On-cloud Agile Data Lake, using HDP.

Other combinations could be used, based on the actual project requirements, application architecture and topology.