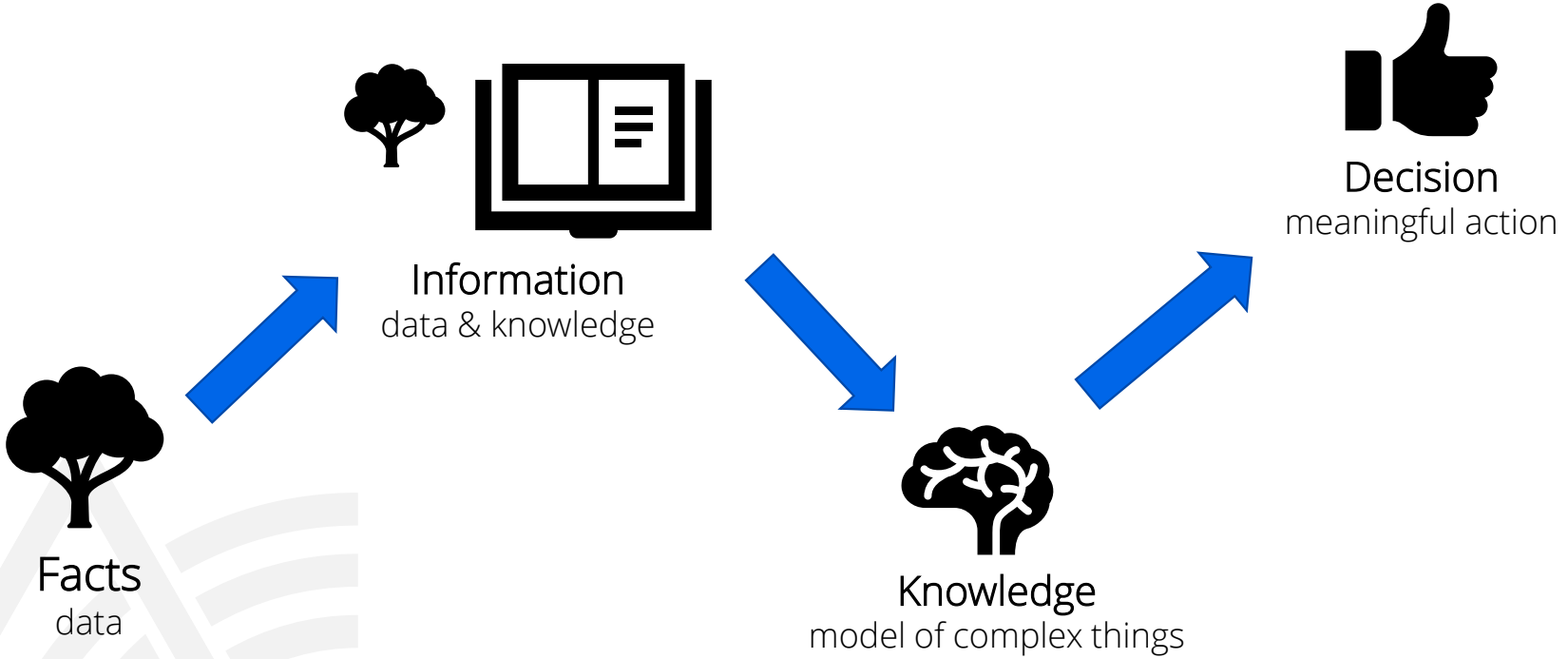
Realtime AI Pipelining

Praxiserprobte Architektur einer modernen Analytics Plattform

Fabian Hardt

Information model



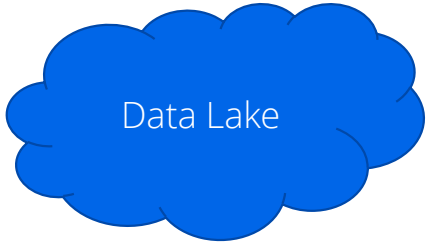
Agenda

1. Data Lake
2. Data Lab / Data Factory / AI Pipeline
3. Data Structures and Metadata
4. Demo
5. Summary

Data Lake

- Purpose
- Requirements
- Architecture

Data Lake - Purpose



- Primary task: Centralized data provisioning
  - All data of a company is collected in one centralized data platform
  - Also external data is collected (social-media, weather, stock market prices, etc.)
- Similar to DWH Systems, integration of different data sources
- Store Raw data without knowing the intended purpose
  - Structured data
  - Unstructured data
- Suitable for very large amounts of data
  - Mostly implemented with big data technologies

Data Lake – Enterprise Requirements

- Usability of stored Data
  - Searchability of data
  - Evaluability of data
  - Availability of data

- Data Security / Data protection regulations
  - Authentication of users
  - Authorization of users

} Users should only see the data that they're allowed to see

- Data Governance
  - Quality Assurance of data
  - Governs the compliance requirements

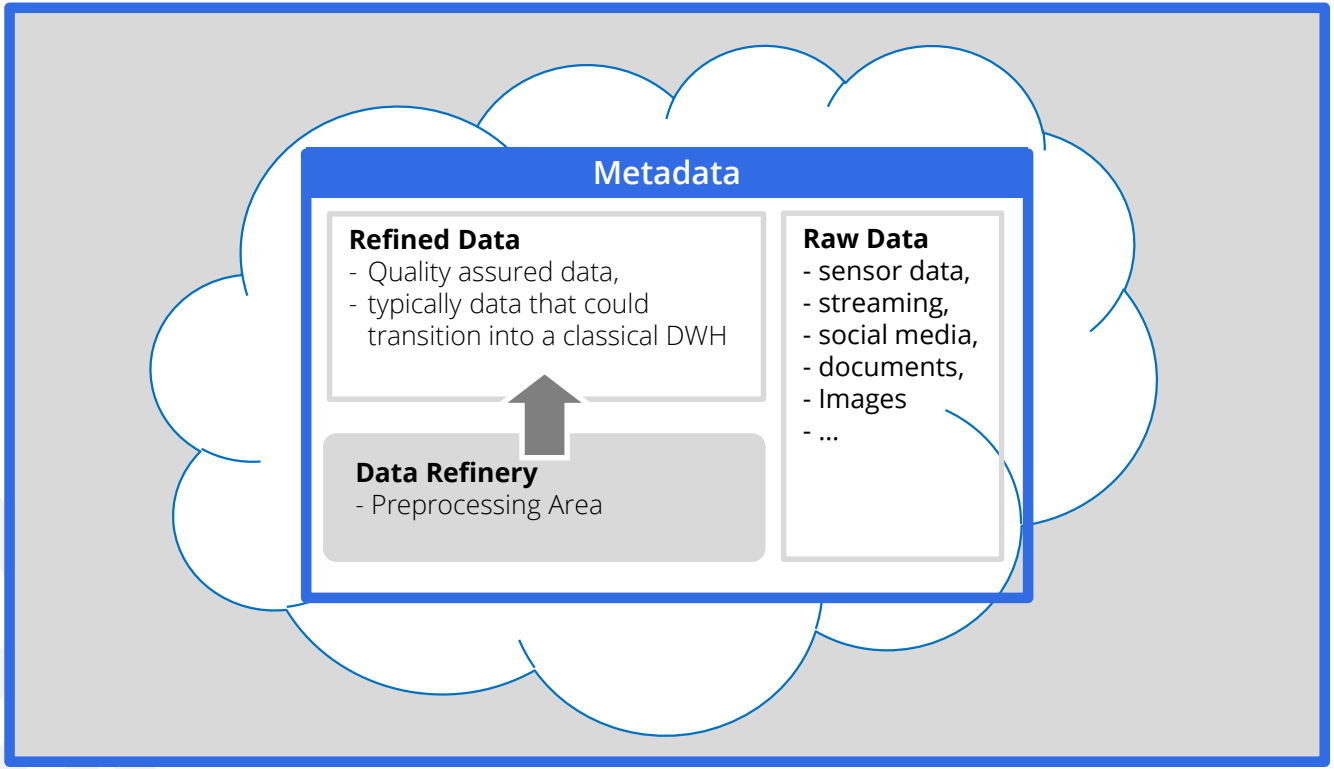
Data Lake – Enterprise Requirements implementation

- Metadata is used to search and evaluate Data
  - All attributes associated with stored data is metadata
- Metadata is used to control user access
  - Basis for access policies
  - Give User access to data based on attributes in metadata
  - Key to find and manage data stored in the Data Lake (e.g. for housekeeping)
- Metadata helps in quality assurance and housekeeping through the ability to search and evaluate
  - Key to find and manage data stored in the Data Lake (e.g. for housekeeping)

Data Lake - Architecture

- Structured in several architectural devided parts
  - Raw Data section
  - Data Refinery
  - Refined Data section
  - Metadata Section
- Technical implementation example:
  - Hadoop Plattform – HDFS as Landing Zone for incomming data
  - Spark Jobs / Kafka Streams – data processing in batch or streaming
  - Hive – store refined data with specific schema
  - Elasticsearch – store metadata with extensible schema
- No Dataloss in case of source system changes– Schema on Read

Core components of a Data Lake



Data Lab - Analytics Sandbox

- Purpose
- Requirements
- Architecture

Data Lab - purpose

- Develop processes and algorithms to gain data insights

- Development Area for Data Scientists
  - Allows Data Scientist to install and use any software of their choice
  - Allows Data Scientist experimental processes with external data
  - Allows Data Scientist to manage all data and processes

Data Lab - Requirements

- Should give all freedoms to Data Scientist
  - Use all kinds of technologies and software (in container)
  - Use data as close as possible to real enterprise data
  - Use data from external sources
- Should match all Requirements from a Software Development Environment
  - Software artifact should be runnable in Data Factory
- Should prevent Data Scientist from seeing data he should not see
- Should ensure data lake does not get messed up
- Should ensure even resource hungry processes do not influence the data lake

Data Science vs. Enterprise

Data Lab

- Looking for business insights in a huge amount of data
  - Mainly used to train algorithms
- Not a production system
  - no operating team necessary
- Usually no other target groups
  - Often no further authorization is needed
  - Usually only authentication is enabled

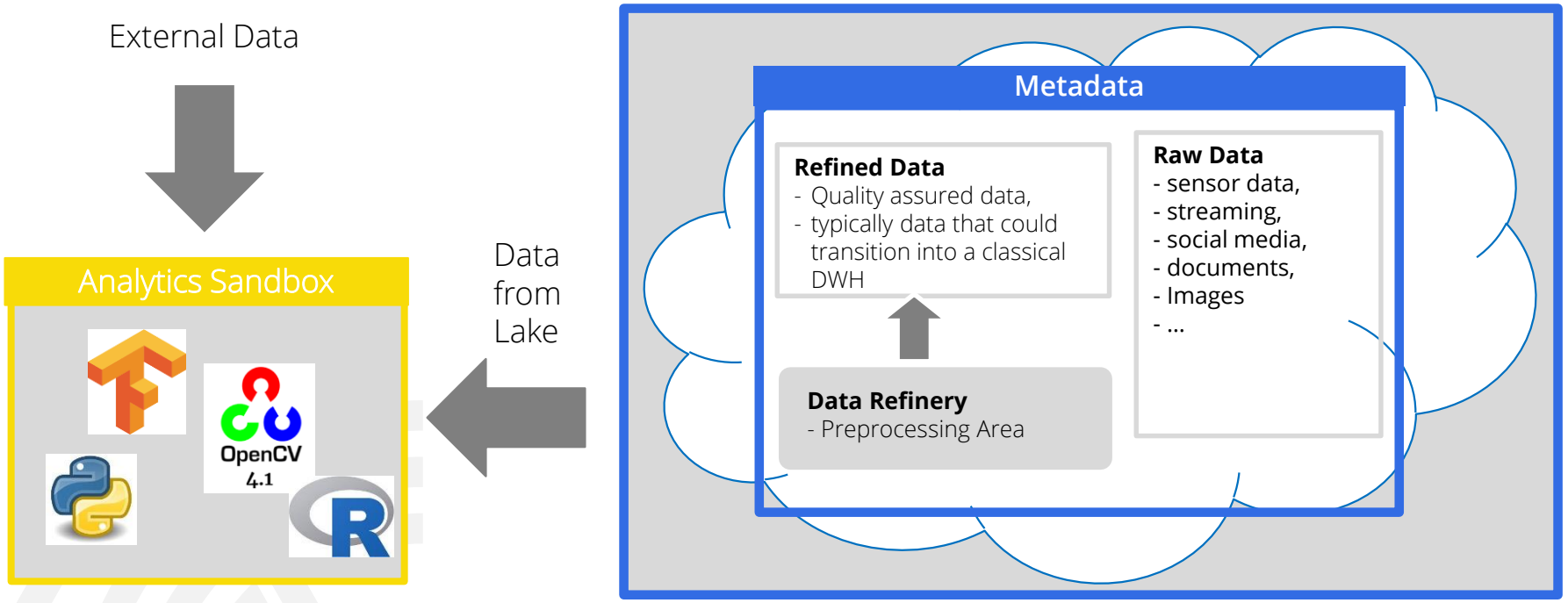
Enterprise Data Lake

- Is very integrated into the daily business
- Production system
  - Typically many target applications are based on this collection of data
  - High availability
  - Backup & Recovery

Data Lab – Requirements implementation

- Data Lab as Sandbox dedicated to Data Scientist
  - No Authorization needed
  - Gives all freedoms to Data Scientist
- Access to Data Lake to use Data from there
  - Only read access at most
  - Usually only working on copies of data from Data Lake
- Sandbox Architecture prevents Data Lab from influencing Data Lake processes

Data Lab - Architecture



Data Factory - purpose

- Generate value from trained algorithms and models from Data Lab
  - Batch Processing
  - Stream Processing
- Enrich Data in Data Lake with insights
  - Similar to data refinement
- Run AI processes on all data where value can be generated

Data Lab

- Explorative approach
- Insights generating
- Work with production near data
- Use Sandboxes
  - Data Scientists
  - Experten der Fachabteilung

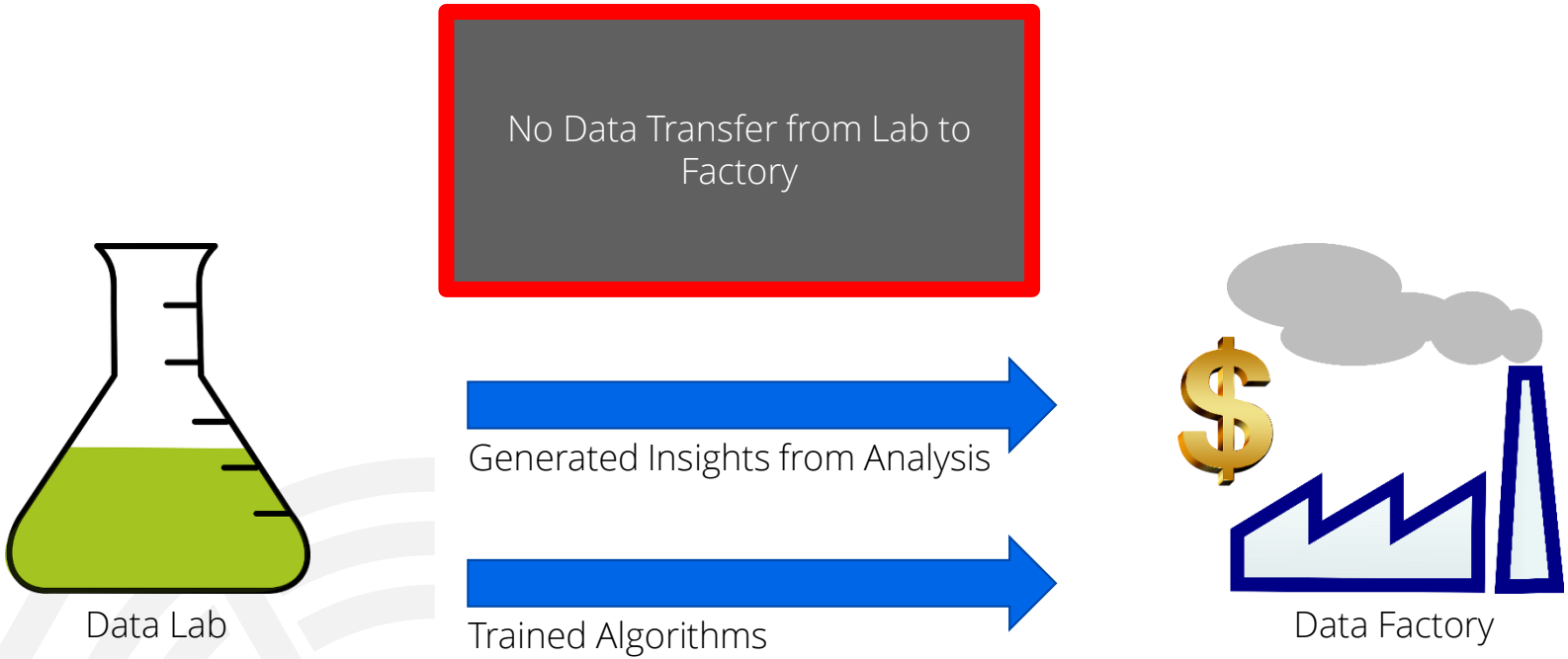
- Goal: Train models and algorithms

Data Factory

- Generating value from trained models and algorithms
- authomatic processing of data from Data Lake
- Further development like in other software development products

- Makes use of trained models and algorithms

Deployment from Data Lab to Factory



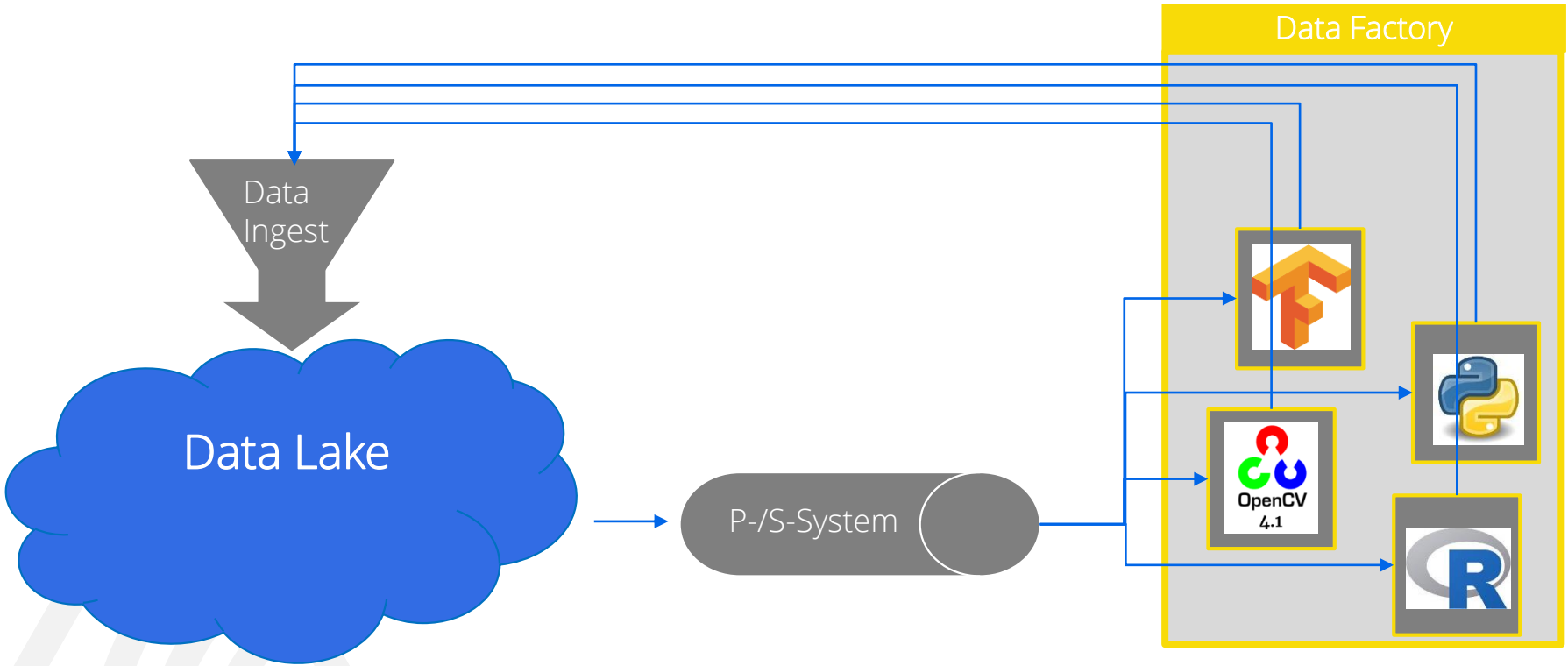
Data Factory - Requirements

- New models and algorithms should be integrated easily
- Established algorithms should be updated easily
- Found insights should be integrated in Data Lake to be evalueable
  - Challenge to design extensible data model
- Single Data Factory components should not influence ohter Data Lake processes
- Must be complient conform

Data Factory – Requirements implementation

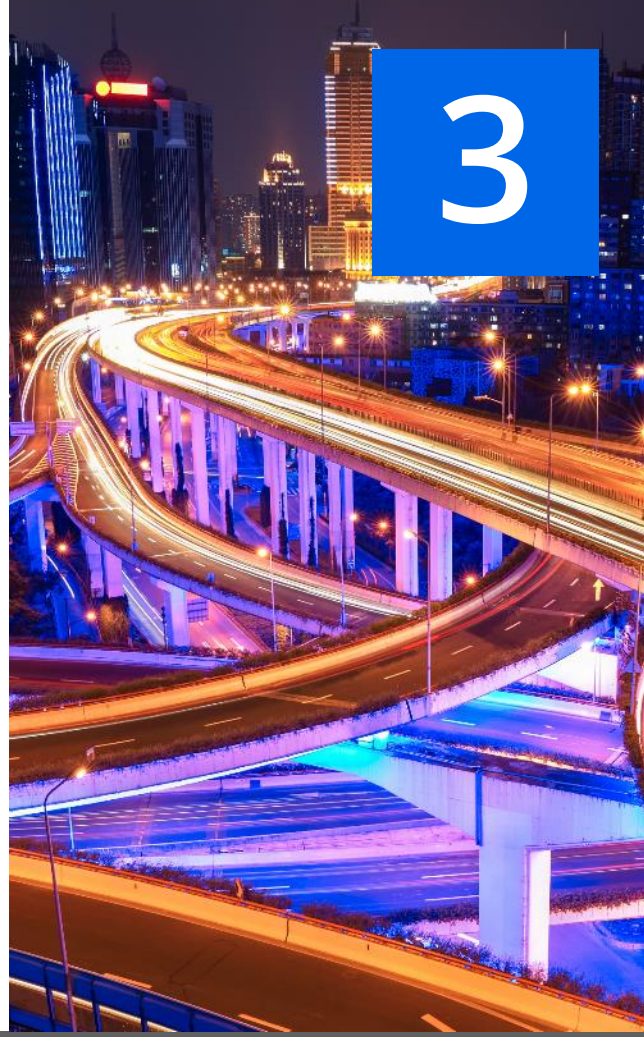
- AI processes and Models run on dedicated Hardware
  - Necessary due to performance (often GPU hardware is needed)
  - No influencing on other Data Lake processes
- Get feed from Data Lake via information Hubs
  - Kafka or other publish-subscibe technologies for example
  - Each process get/filters the data it needs
- Insert insights back into Data Lake to be processed as any other data inserted
  - Found Insights from one process might be used by other processes

Data Factory - Architecture



Data Structures and Metadata

- Schema on Write vs. Schema on Read
- Metadata Structure in Data Lake



Schema on Write vs. Schema on Read

- Schema on Write
  - Even during the write operation, the target (scheme) must be clearly defined.
  - Example: There is a table structure in which suitable data must be inserted.

- Schema on Read
  - For writing operations the target (scheme) need not be known.
  - "First throw everything in it. At a later time look how to get it out again "
  - That is: Only for the read operation, the big data source (the scheme of this) must be known.

Schema on Write

- „old school Data Warehouses” (relational DB)
- ⊕ Oriented at the customer requests
- ⊖ Not very agile, changing source systems has an impact on import ETL
- ⊖ Data loss if ETL process is not changed in time

Schema on Read

- Typical Big Data paradigm
- ⊕ Schematization of data only when reading it from the data lake
- ⊕ No loss of data when writing it into the data lake, despite unannounced changes to the source system
- Architectural part in Data Lake: Raw Data Area, Data Refinery

Data Structures in Data Lakes

- Metadata for security and governance should be schema on Write
  - Consistent reliable implementation of security rules
- New information and insights might not fit into predefined schema
- Evaluation Process might be undefined at time of inserting

  → Schema on Read might be best approach

- How can the new information be found?

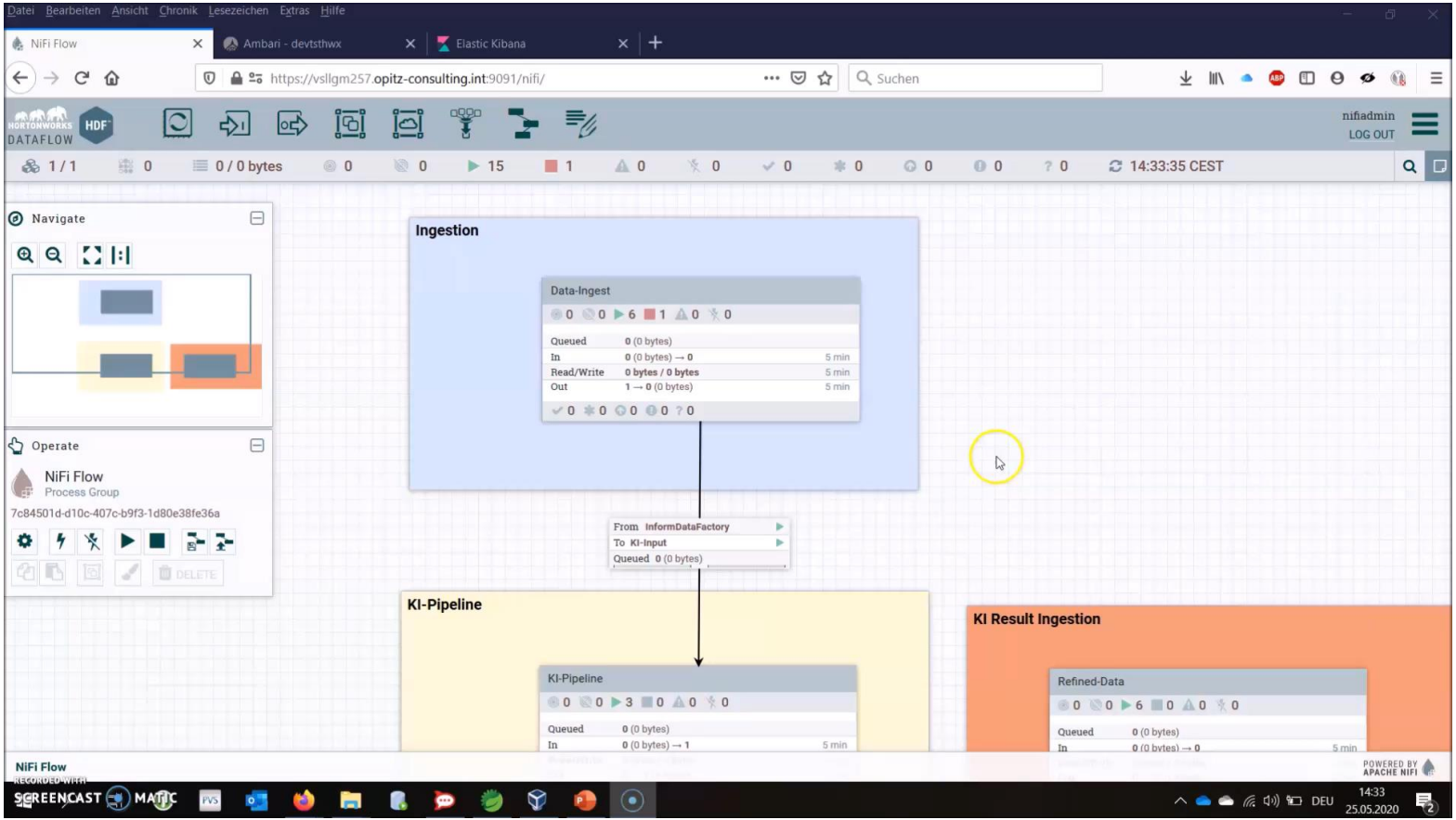
How to store new Insights?

- Most important: same unique identifier for related Data
- Not important: Way to analyze Data
  - Is defined later on

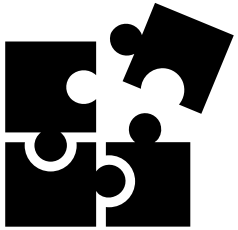
- Data should be stored without loosing information by applying schema
  - Easiest way often is the best
    - Key-Value Stores (Hbase etc.)
    - Elasticsearch with dynamic mapping
    - Specific store per AI process

Demo

- Time for a quick demonstration…


NiFi Flow
Ambari - devtsthwx
Elastic Kibana
https://vsllgm257.opitz-consulting.int:9091/nifi/
Suchen
nifiadmin
LOG OUT
14:33:35 CEST
Navigate
Operate
NiFi Flow
Process Group
7c84501d-d10c-407c-b9f3-1d80e38fe36a
DELETE
Ingestion
Data-Ingest
Queued 0 (0 bytes)
In 0 (0 bytes) → 0 5 min
Read/Write 0 bytes / 0 bytes 5 min
Out 1 → 0 (0 bytes) 5 min
From InformDataFactory
To KI-Input
Queued 0 (0 bytes)
KI-Pipeline
KI-Pipeline
Queued 0 (0 bytes)
In 0 (0 bytes) → 1 5 min
KI Result Ingestion
Refined-Data
Queued 0 (0 bytes)
In 0 (0 bytes) → 0 5 min
NiFi Flow
RECORDED WITH SCREENCAST MATIC
POWERED BY APACHE NIFI
DEU
14:33
25.05.2020

Summary



- Separate Data Lake Components
  - Decupled Data Lake
  - More flexibility
- Generify Input and Output of Data Lakes
  - Flexibility to add new Components
- Separate Development from Production Processes
- Worry about security and governance concerns
  - Apply schema on write

Vielen Dank für Ihr Interesse, wir freuen uns auf den gemeinsamen Austausch mit Ihnen!

Fabian Hardt

Senior Consultant,
Business Intelligence & Analytics

Kirchstraße 6
51647 Gummersbach

Fabian.Hardt@opitz-consulting.com
+49 (0) 2261 6001-1045

WWW.OPITZ-CONSULTING.COM

@OC_WIRE

OPITZCONSULTING

opitzconsulting

opitz-consulting-bcb8-1009116