Big DataBase Management System

Alberto Abelló

(with slides from Sergi Nadal, and Oscar Romero)

Universitat Politècnica de Catalunya

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

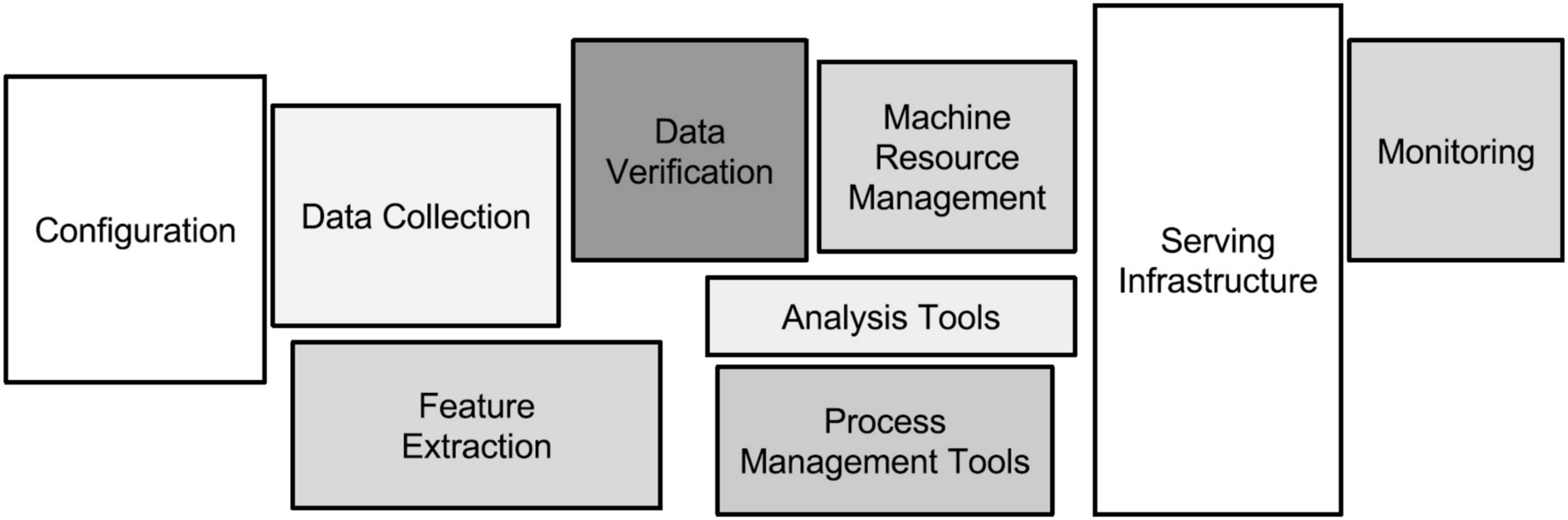
DTIM www.essi.upc.edu/dtim

Contents

- Problem definition
- Data Science specific tasks
- New architectural patterns
  - Storage
  - Processing
- Big DBMS components view
- Big DBMS in use
  - Metadata management

Problem definition

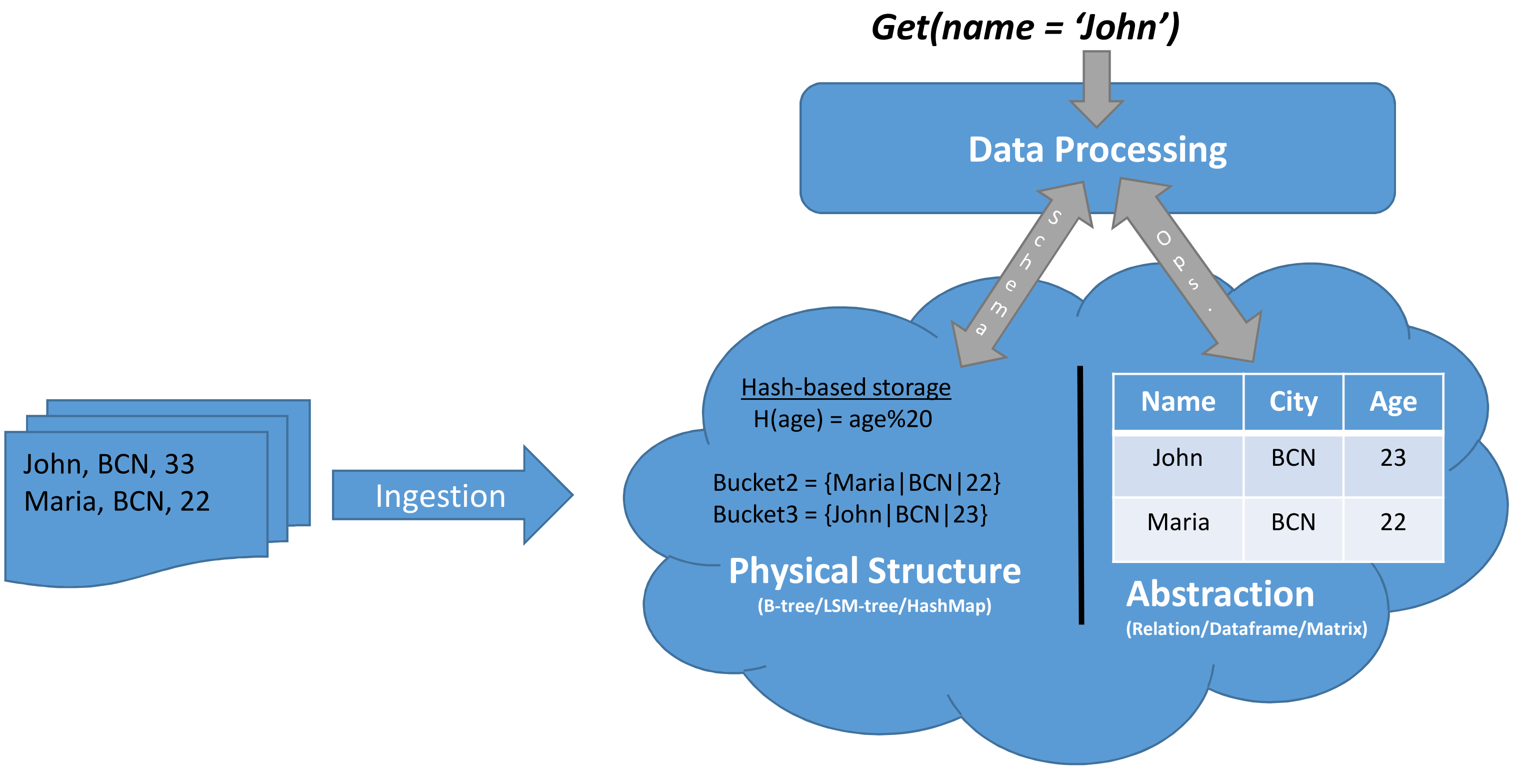
Vast and Complex surrounding infrastructure



- ML code is just a small box in the middle
  - … with a lot of plumbing around it

D. Sculley te al.

Data Management (I)



Name	City	Age
John	BCN	23
Maria	BCN	22

Data Management (II)

Data management refers to the functionalities a DBMS must provide:

- Ingestion: means provided to insert /upload data
  - E.g., ORACLE SQL*Loader
- Storage: format/structures used to persist data
  - E.g., hash, B-tree, heap file
- Modelling: arrangement of data within the available structures
  - E.g., normalization, partitioning
- Processing: means provided to manipulate data
  - E.g., PL/SQL
- Querying/fetching: means provided to allow users to retrieve data
  - E.g., SQL, Relational Algebra

In Big Data settings, they are the same concepts but assuming NOSQL underneath

1. Typically, a distributed system
2. Possibly with an alternative data model to the Relational one
3. Implementing ad-hoc architectural solutions

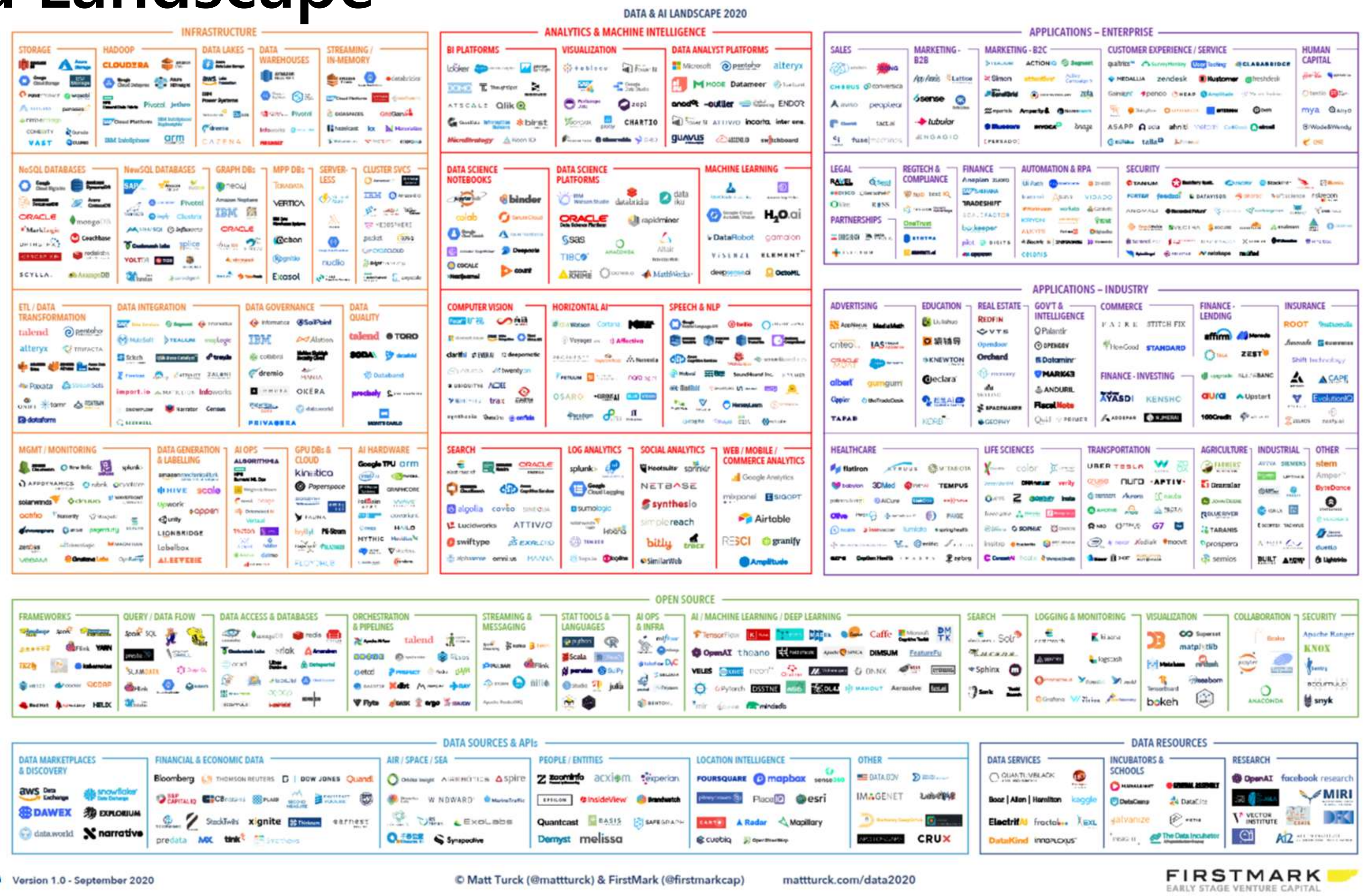
Big Data Architectures

- Question the main principles of traditional DB architectures
  - Data can grow beyond limits requiring scale out (a.k.a. Volume)
  - Data is not necessarily persisted (a.k.a. Velocity)
  - Data structure is neither known a priori, nor fixed (a.k.a. Variety and Variability)
- Use new trendy technological features
  - Primary indexes to implement the global catalog
    - Distributed Tree
    - Dynamic Hashing
  - In-memory processing
  - Columnar block iteration: vertical fragmentation + fixed-size values + RLE compression
    - Heavily exploited by column-oriented databases
    - Good for read-only workloads
  - Sequential reads for large workloads
    - Take the most out of databases by boosting sequential reads
      - Enables pre-fetching
      - Option to maximize the effective read ratio (by a good DB design)
        - Key design
- Implement from scratch the whole stack
  - Ingestion, Storage, Modeling, Processing, and Querying

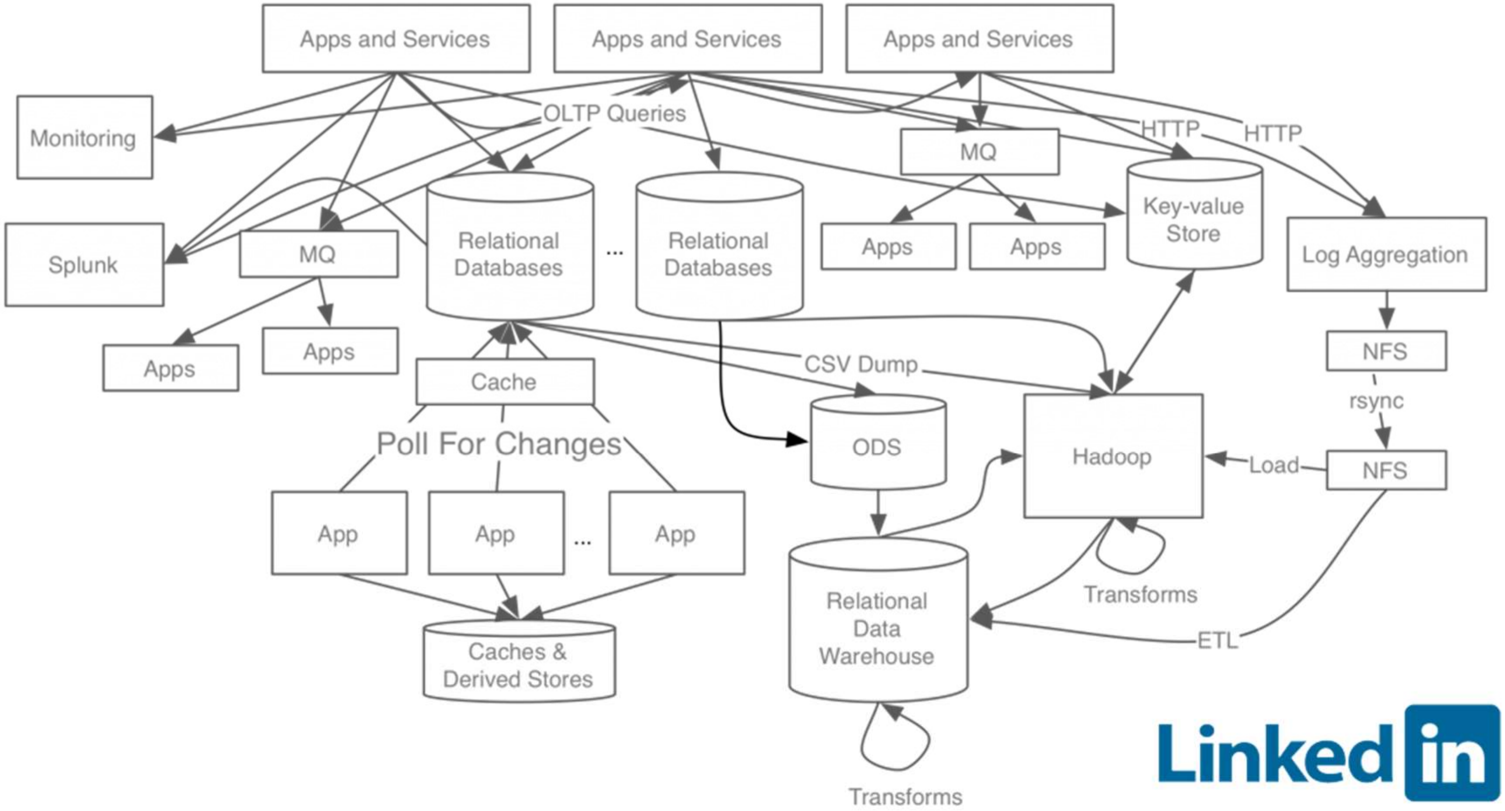
The Multi-Project Approach

- The DBMS tasks are spread over different systems
  - Independent
  - Heterogeneous
- Hadoop is a paradigmatic case:
  - Storage: HDFS + Hbase
  - Modeling: HCatalog
  - Ingestion: Sqoop
  - Processing: Spark
  - Querying: Spark SQL

Big Data Landscape



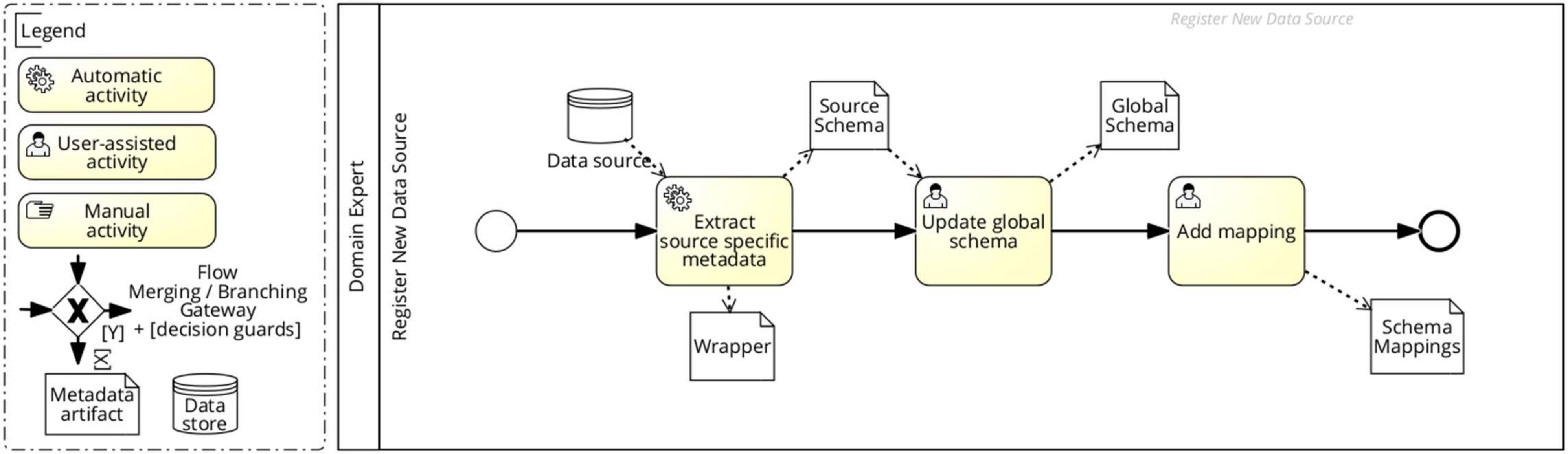
Spaghetti architecture



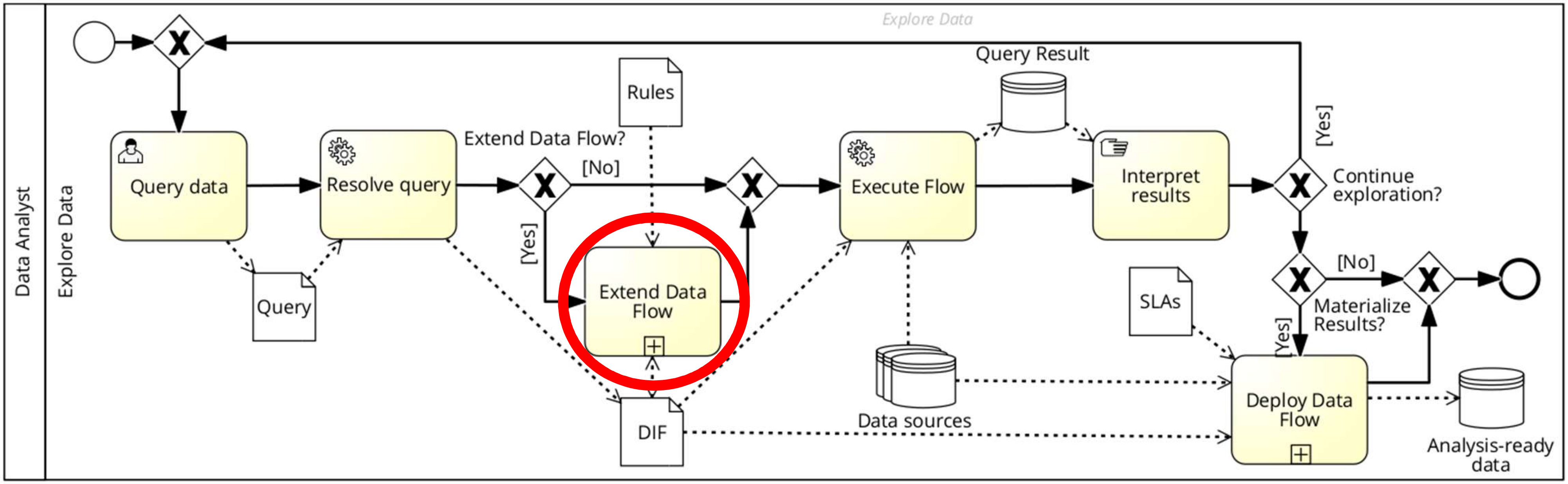
https://www.confluent.io/blog/event-streaming-platform-1

Specific data engineering tasks

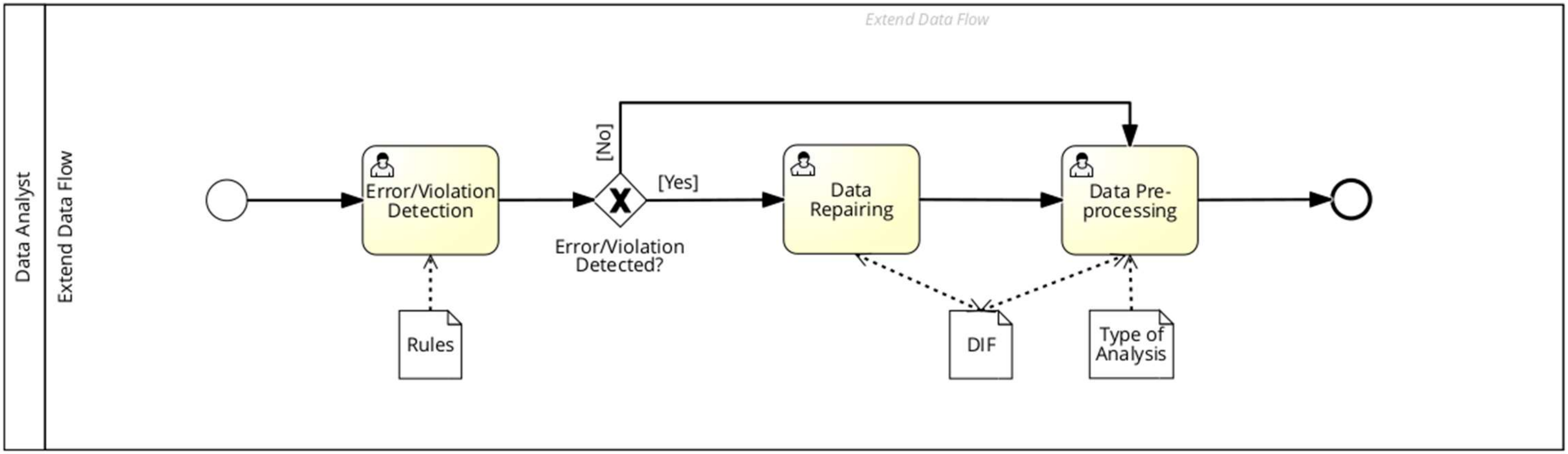
Register new data source



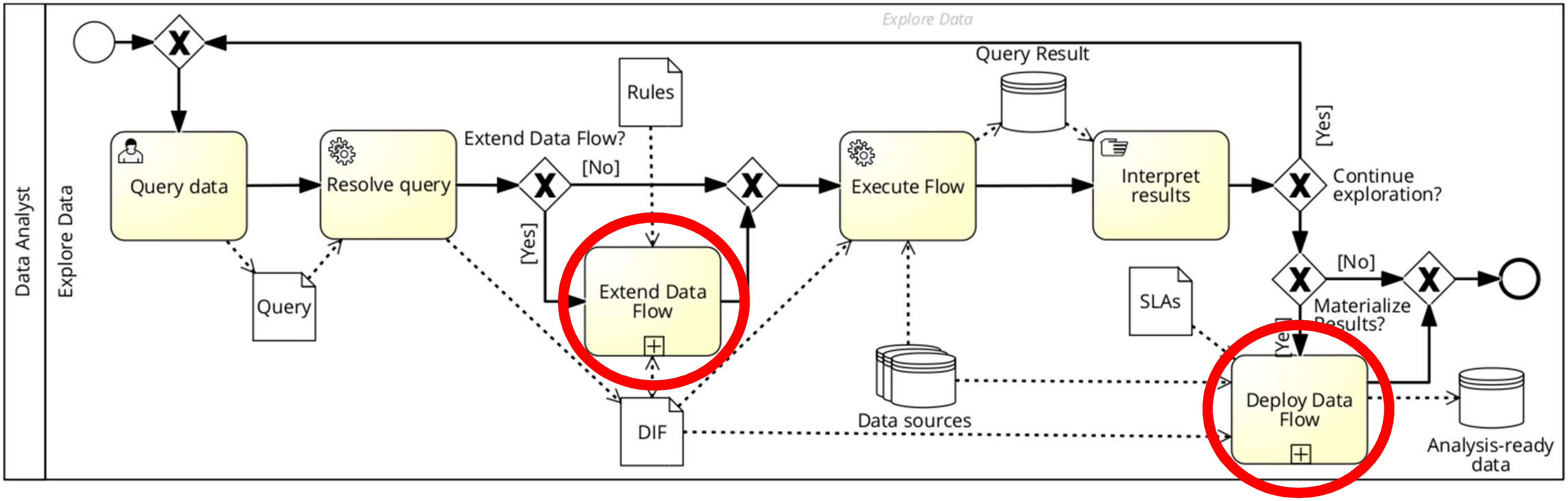
Explore data



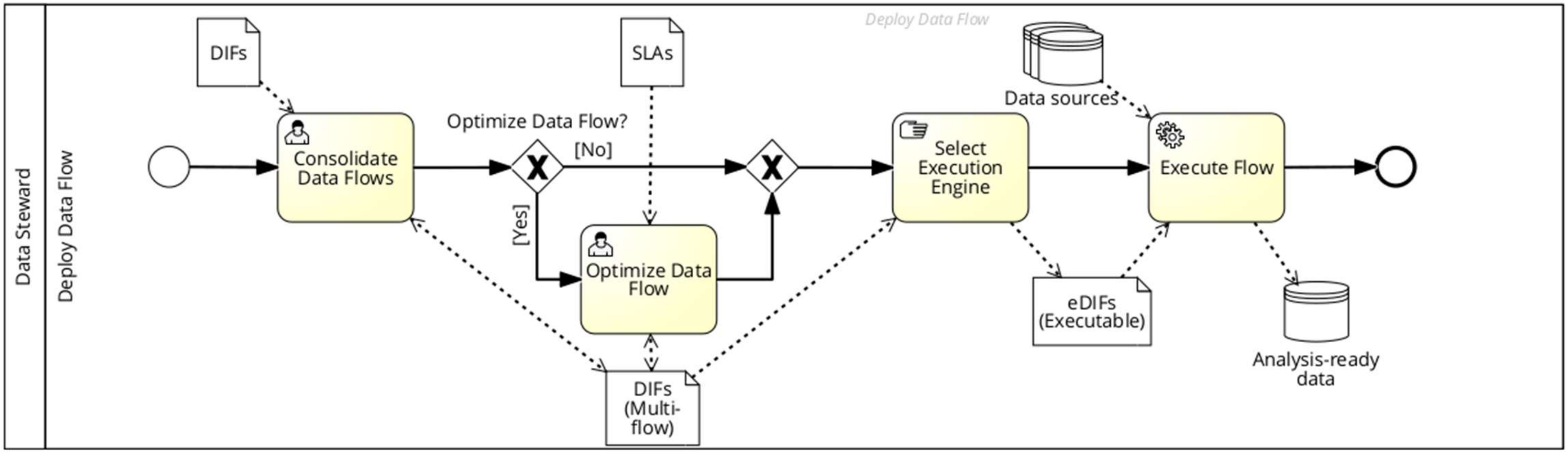
Extend data flow



Explore data



Deploy data flow

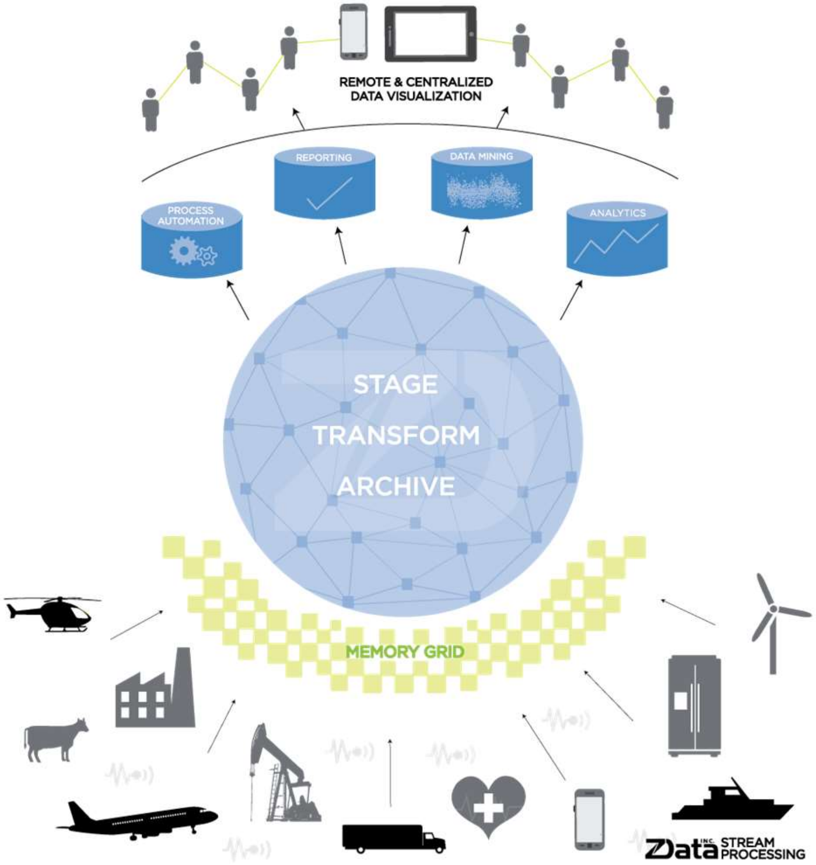


New Storage Architectural Pattern

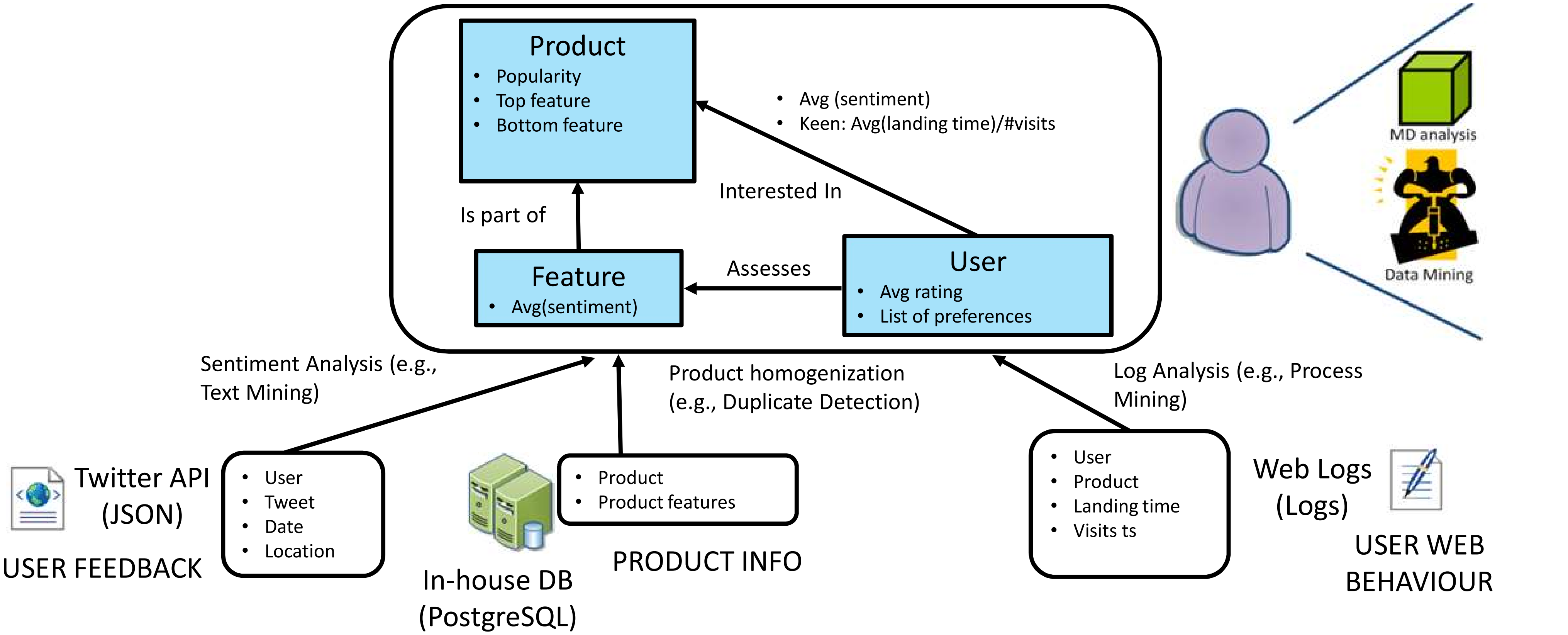
From data warehousing to data lakes

The Data Lake

- Idea: Load-First, Model-Later
- Modelling at load time restricts the potential analysis that can be done later (Big Analytics)
- Store raw data and create on-demand views to handle with precise analysis needs



Model-First (Load-Later)



Product
- Popularity
- Top feature
- Bottom feature

Is part of

Feature
- Avg(sentiment)

Assesses

User
- Avg rating
- List of preferences

Interested In

- Avg (sentiment)
- Keen: Avg(landing time)/#visits

Sentiment Analysis (e.g., Text Mining)

Product homogenization (e.g., Duplicate Detection)

Log Analysis (e.g., Process Mining)

Twitter API (JSON)

- User
- Tweet
- Date
- Location

USER FEEDBACK

In-house DB (PostgreSQL)

- Product
- Product features

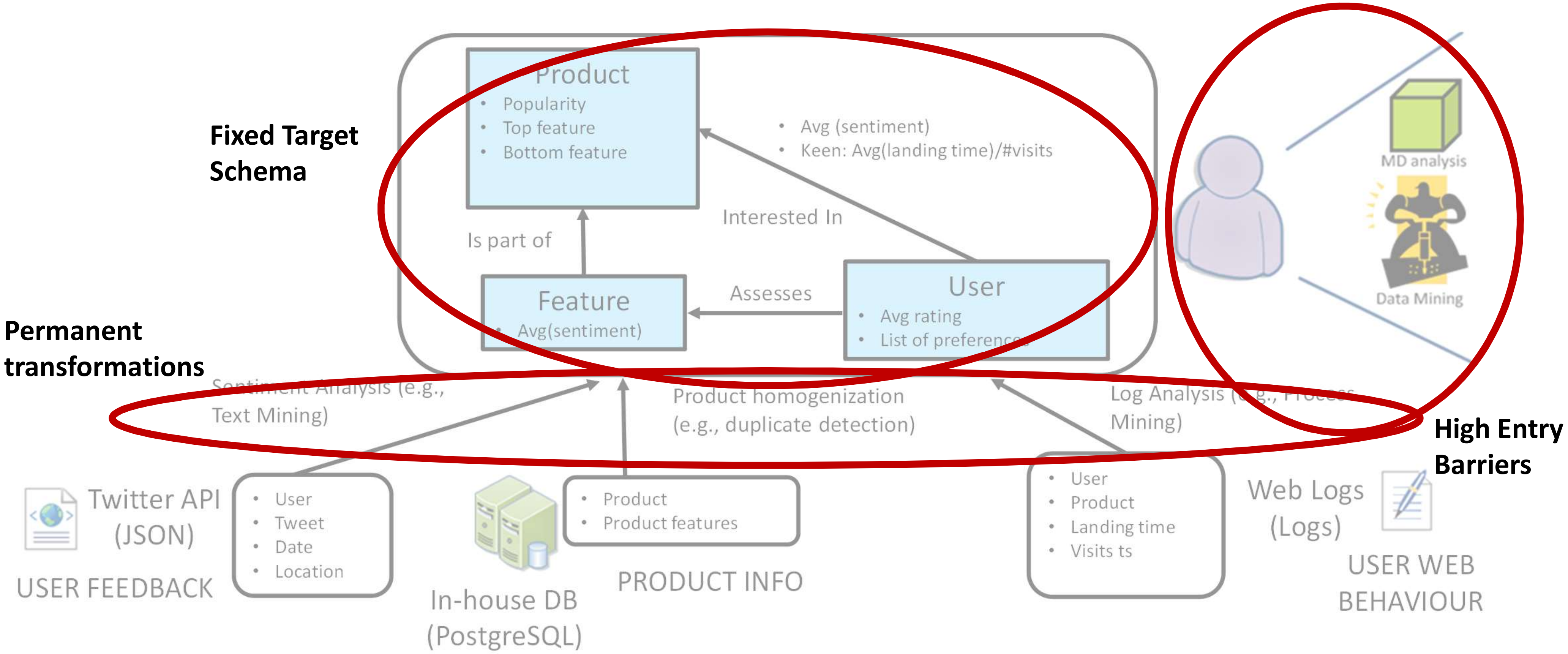
PRODUCT INFO

- User
- Product
- Landing time
- Visits ts

Web Logs (Logs)

USER WEB BEHAVIOUR

Drawbacks of Model-First (Load-Later)



Fixed Target Schema

Permanent transformations

High Entry Barriers

Product
- Popularity
- Top feature
- Bottom feature

- Avg (sentiment)
- Keen: Avg(landing time)/#visits

Interested In

Is part of

Feature
- Avg(sentiment)

Assesses

User
- Avg rating
- List of preferences

MD analysis

Data Mining

Sentiment Analysis (e.g., Text Mining)

Product homogenization (e.g., duplicate detection)

Log Analysis (e.g., Process Mining)

Twitter API (JSON)

USER FEEDBACK

- User
- Tweet
- Date
- Location

In-house DB (PostgreSQL)

- Product
- Product features

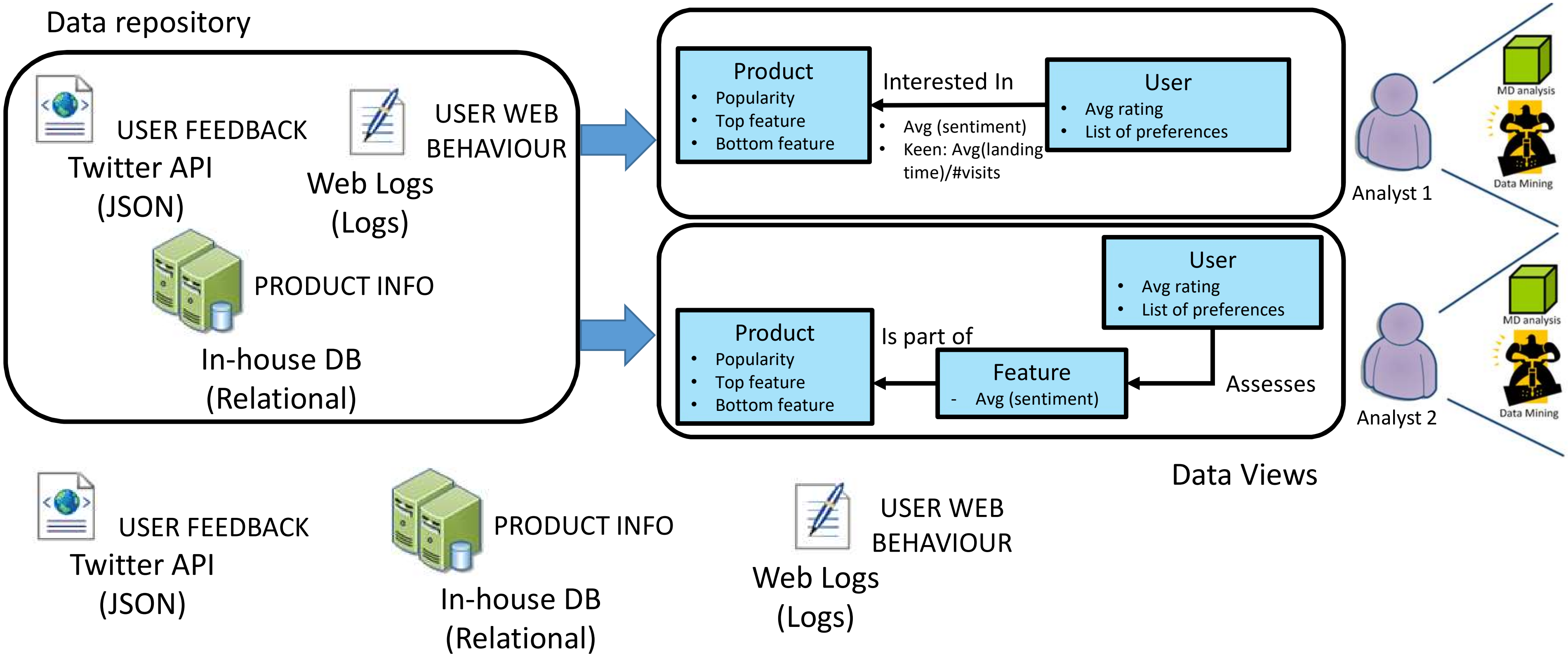
PRODUCT INFO

- User
- Product
- Landing time
- Visits ts

Web Logs (Logs)

USER WEB BEHAVIOUR

Load-First (Model-Later)



Data repository

USER FEEDBACK
Twitter API
(JSON)

USER WEB
BEHAVIOUR
Web Logs
(Logs)

PRODUCT INFO
In-house DB
(Relational)

Product
- Popularity
- Top feature
- Bottom feature

Interested In
- Avg (sentiment)
- Keen: Avg(landing time)/#visits

User
- Avg rating
- List of preferences

User
- Avg rating
- List of preferences

Product
- Popularity
- Top feature
- Bottom feature

Is part of

Feature
- Avg (sentiment)

Assesses

Data Views

Analyst 1

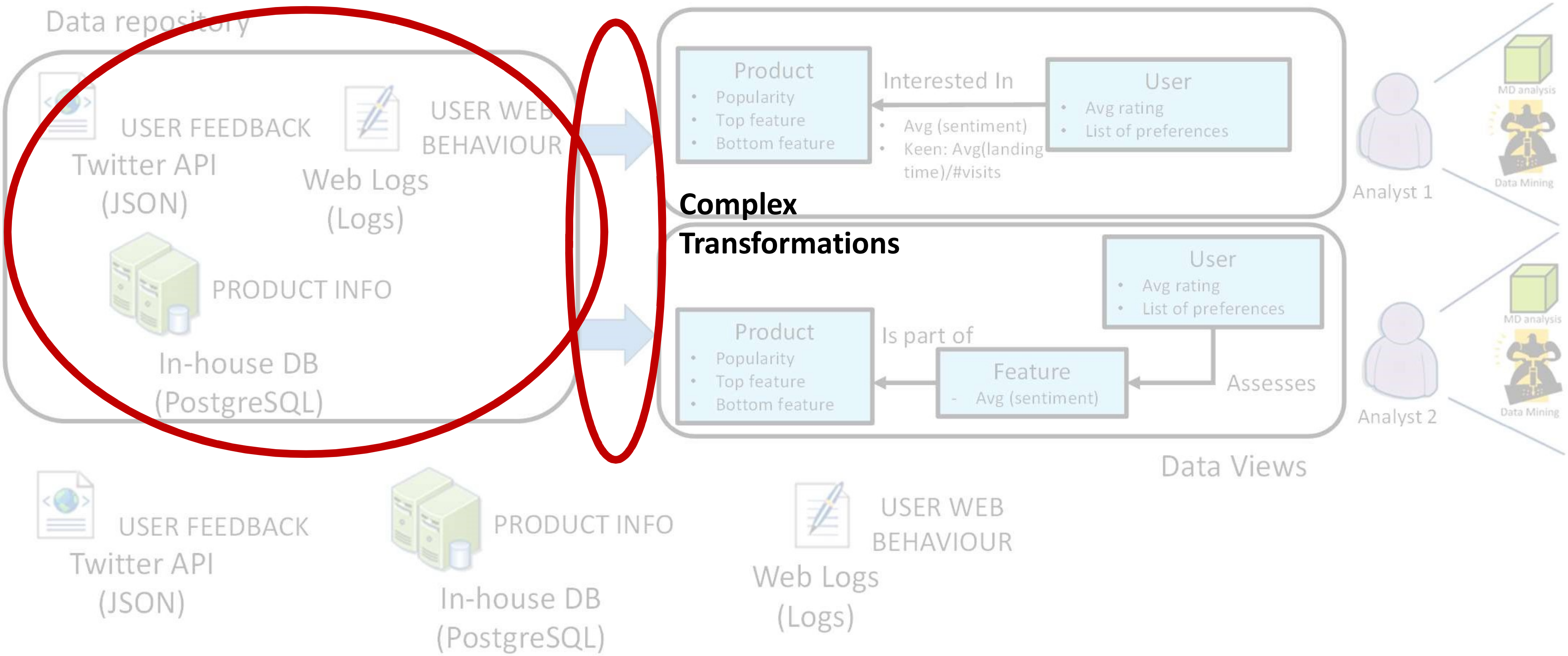
Analyst 2

USER FEEDBACK
Twitter API
(JSON)

PRODUCT INFO
In-house DB
(Relational)

USER WEB
BEHAVIOUR
Web Logs
(Logs)

Drawbacks of Load-First (Model-Later)

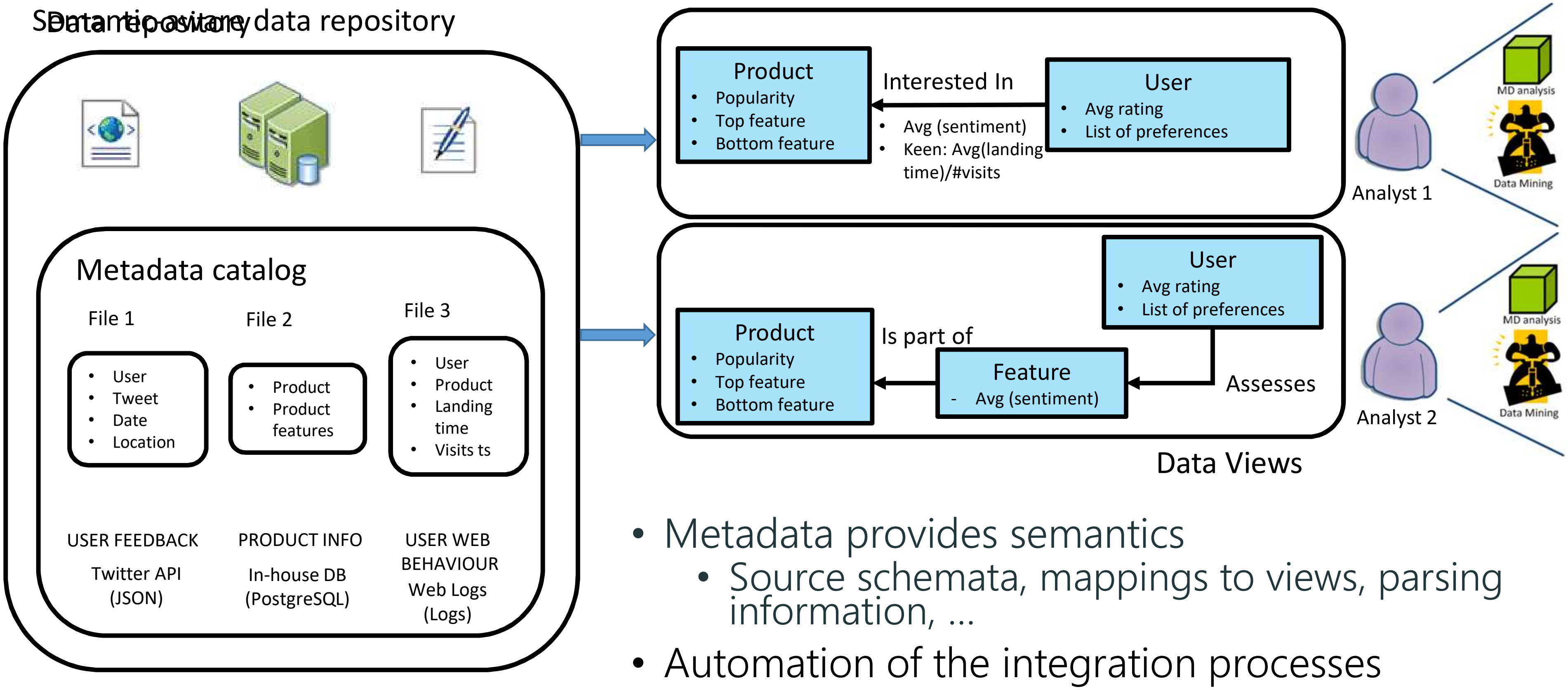


Data Swamp

Complex Transformations

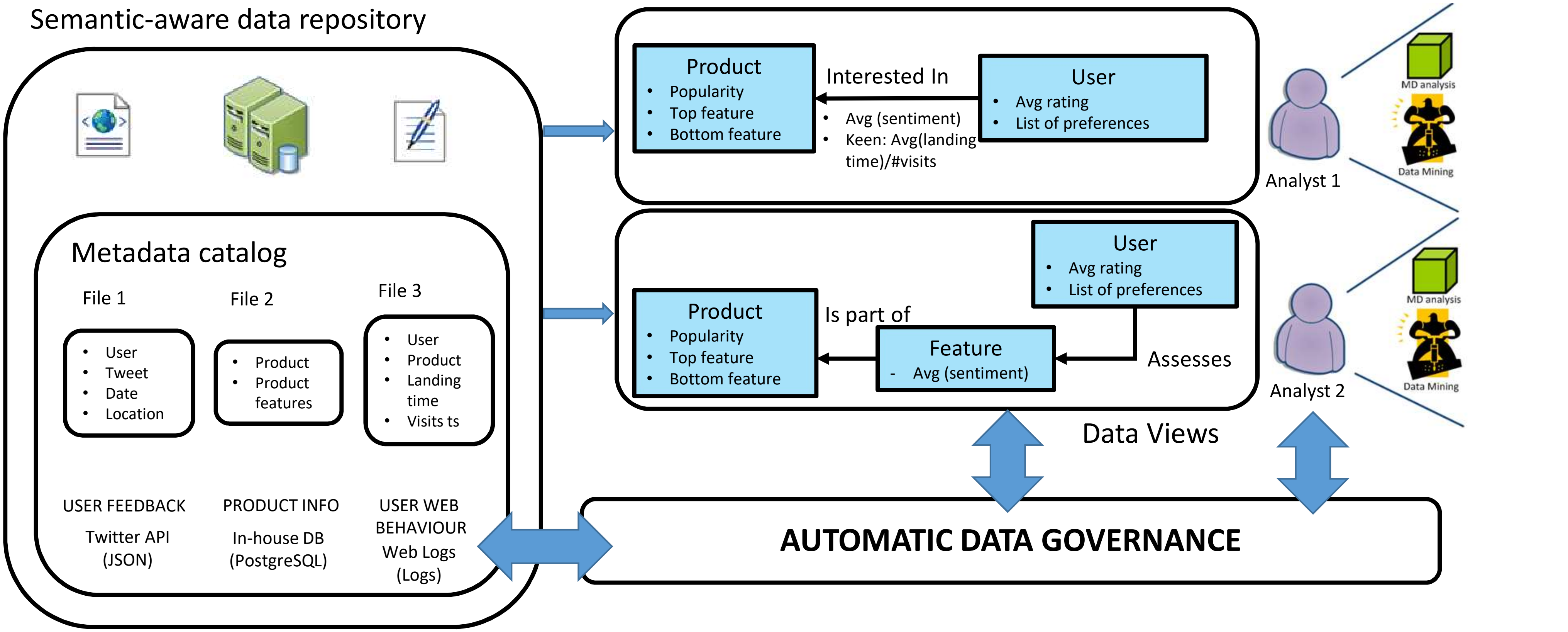
Stonebraker (2014)

Towards semantic-awareness



- Metadata provides semantics
  - Source schemata, mappings to views, parsing information, …
- Automation of the integration processes

From IT-Centered to User-Centered



Semantic-aware data repository

Metadata catalog

File 1
- User
- Tweet
- Date
- Location

File 2
- Product
- Product features

File 3
- User
- Product
- Landing time
- Visits ts

USER FEEDBACK
Twitter API
(JSON)

PRODUCT INFO
In-house DB
(PostgreSQL)

USER WEB BEHAVIOUR
Web Logs
(Logs)

Product
- Popularity
- Top feature
- Bottom feature

Interested In
- Avg (sentiment)
- Keen: Avg(landing time)/#visits

User
- Avg rating
- List of preferences

User
- Avg rating
- List of preferences

Product
- Popularity
- Top feature
- Bottom feature

Is part of

Feature
- Avg (sentiment)

Assesses

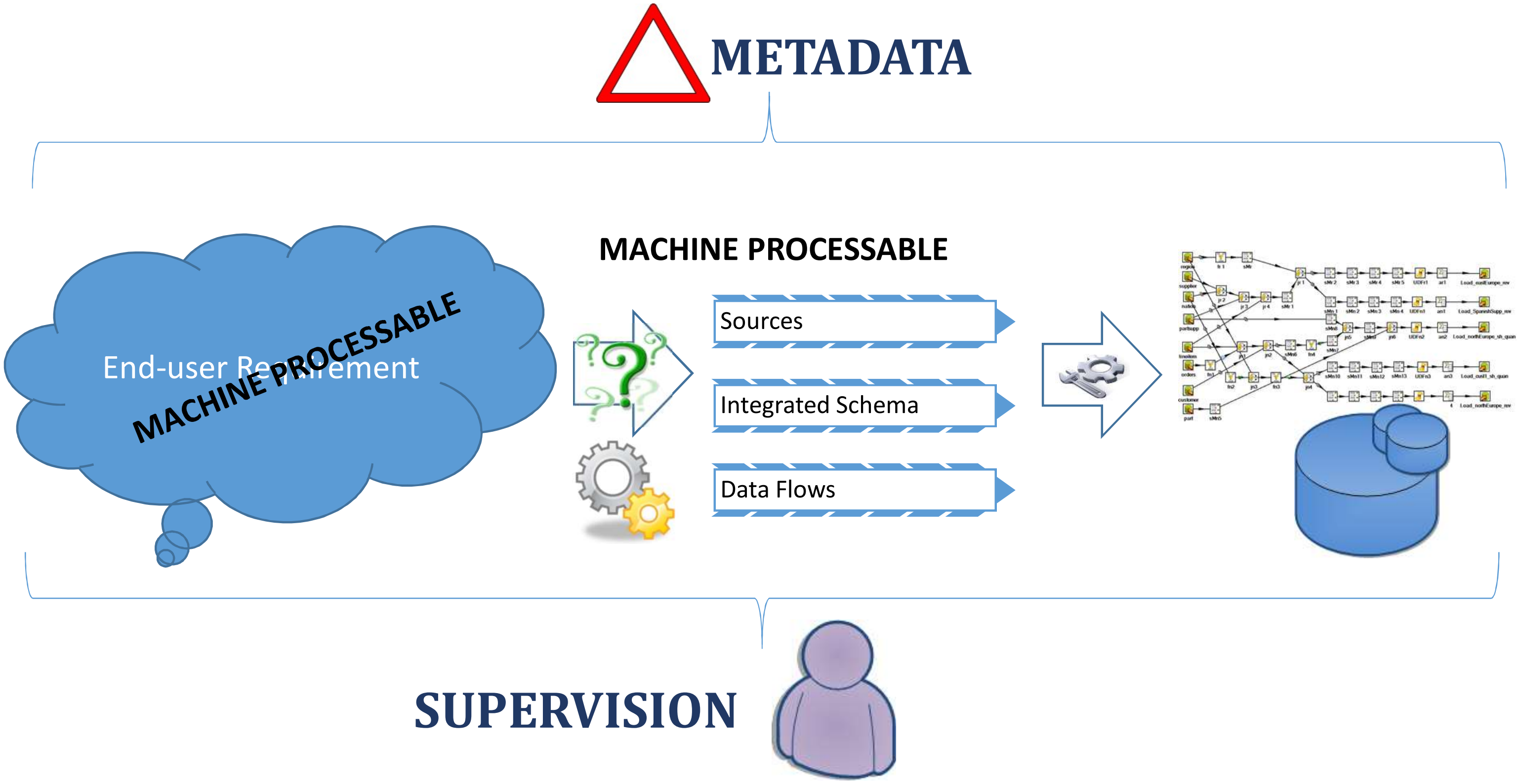
Data Views

AUTOMATIC DATA GOVERNANCE

Analyst 1

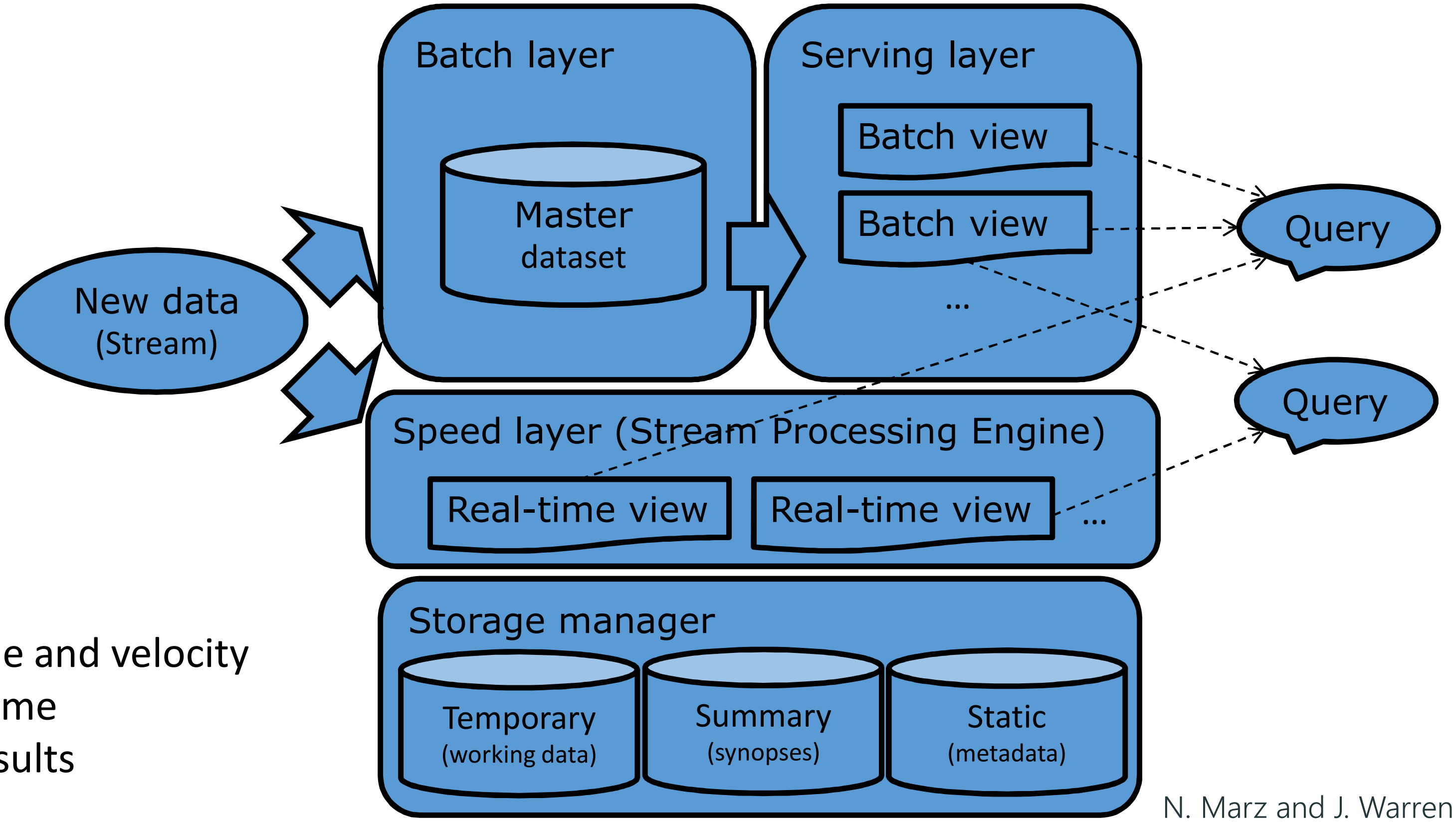
Analyst 2

The Missing Link: Metadata



New Processing Architectural Patterns

λ-Architecture



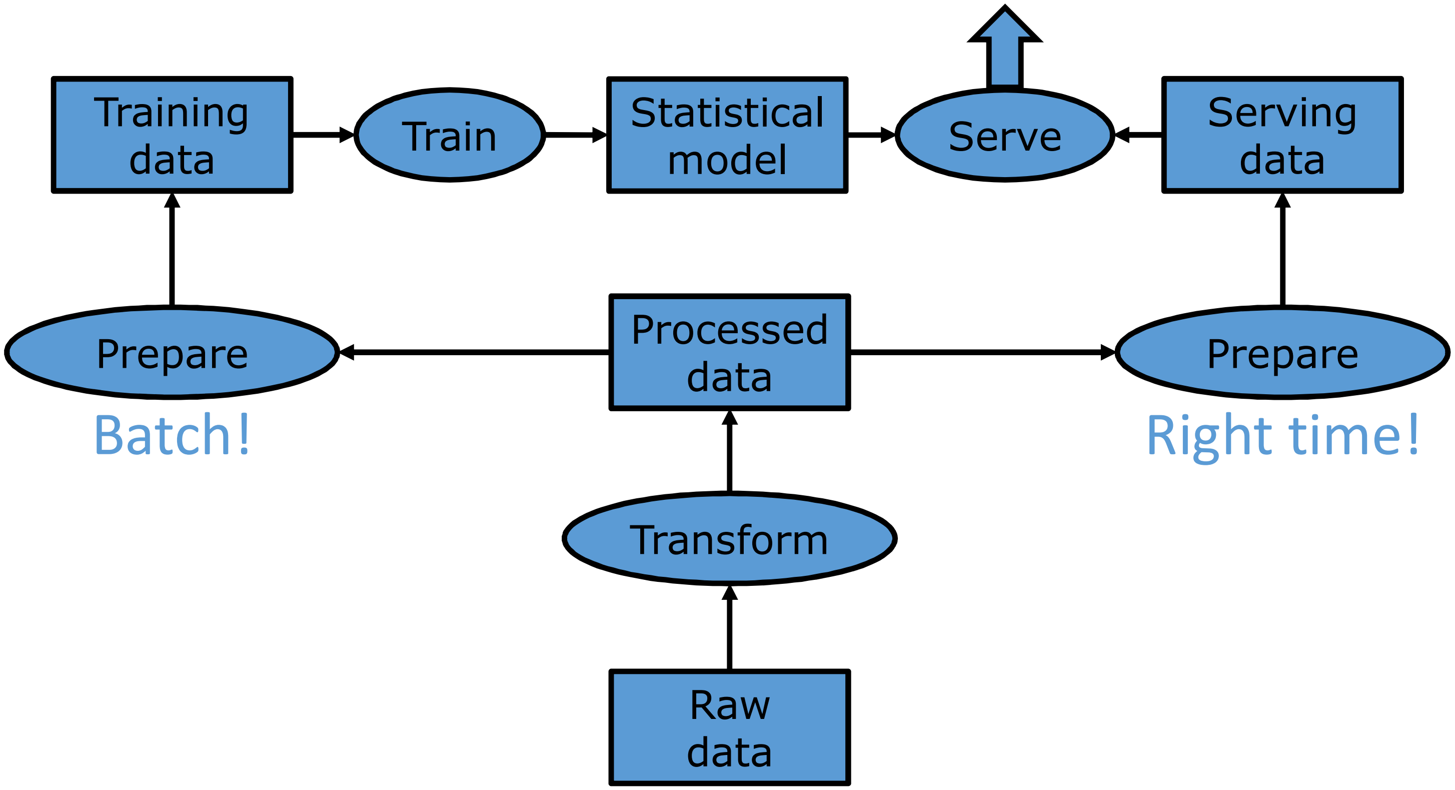
N. Marz and J. Warren

Idea: Accommodate volume and velocity

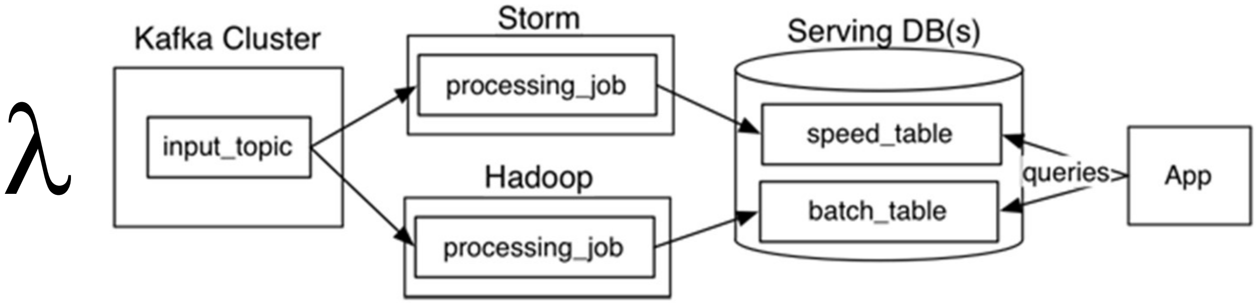
Batch processing vs. Real time

Precise vs. Approximate results

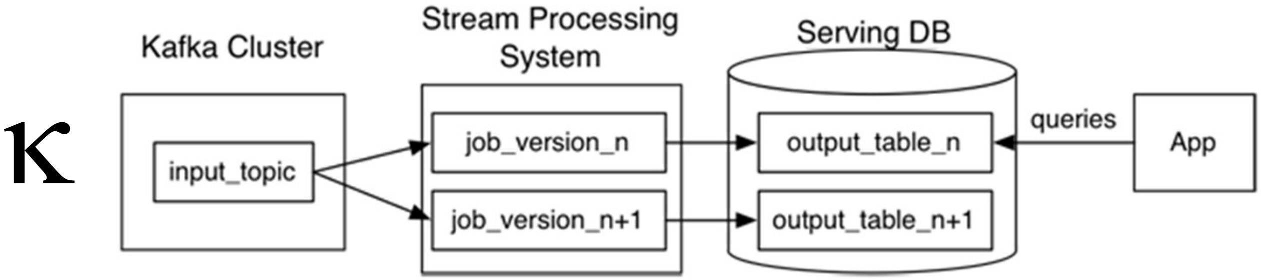
Data-centered architecture



κ-architecture



- Data is all considered to be a never-ending stream

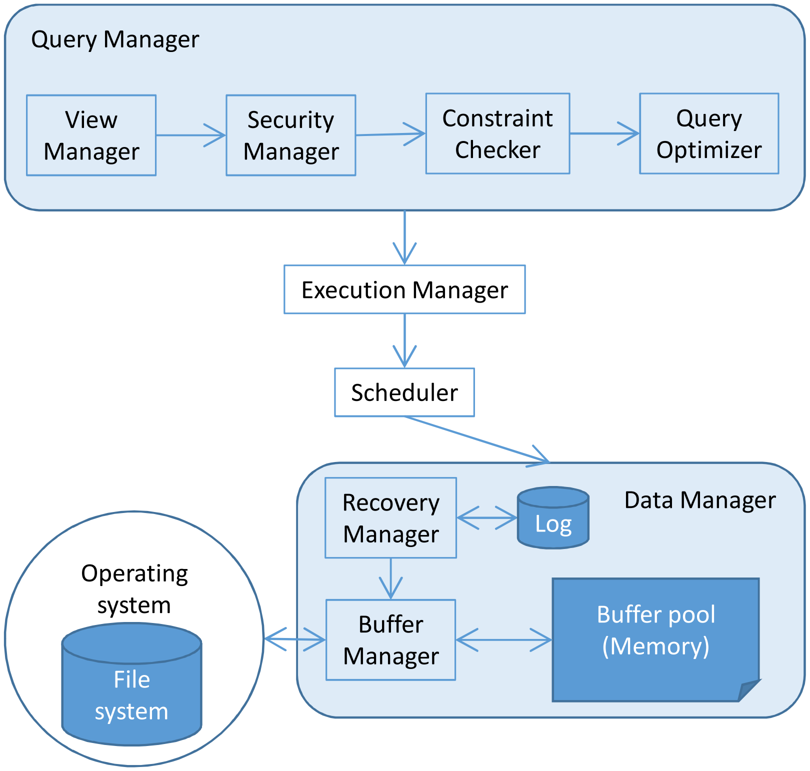


Jay Kreps and O'Reilley

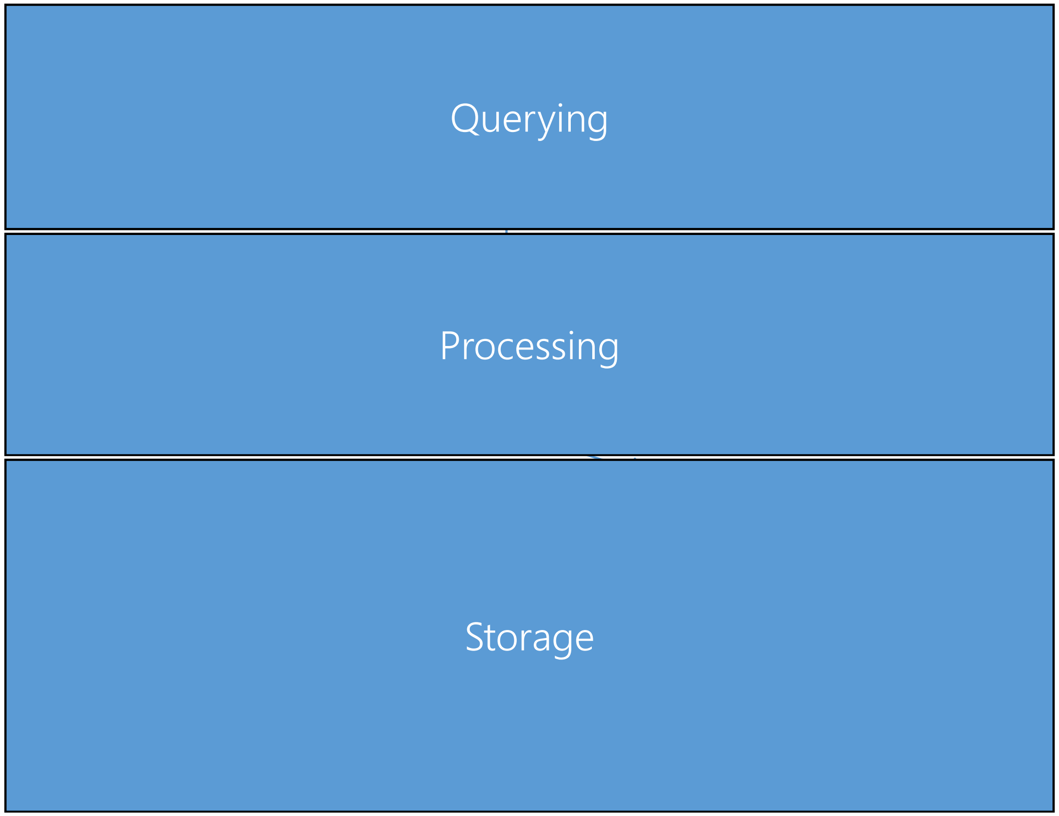
Big Database Management System <u>components view</u>

Tools/Roles and their connections

Centralized DBMS Architecture



Centralized DBMS Architecture

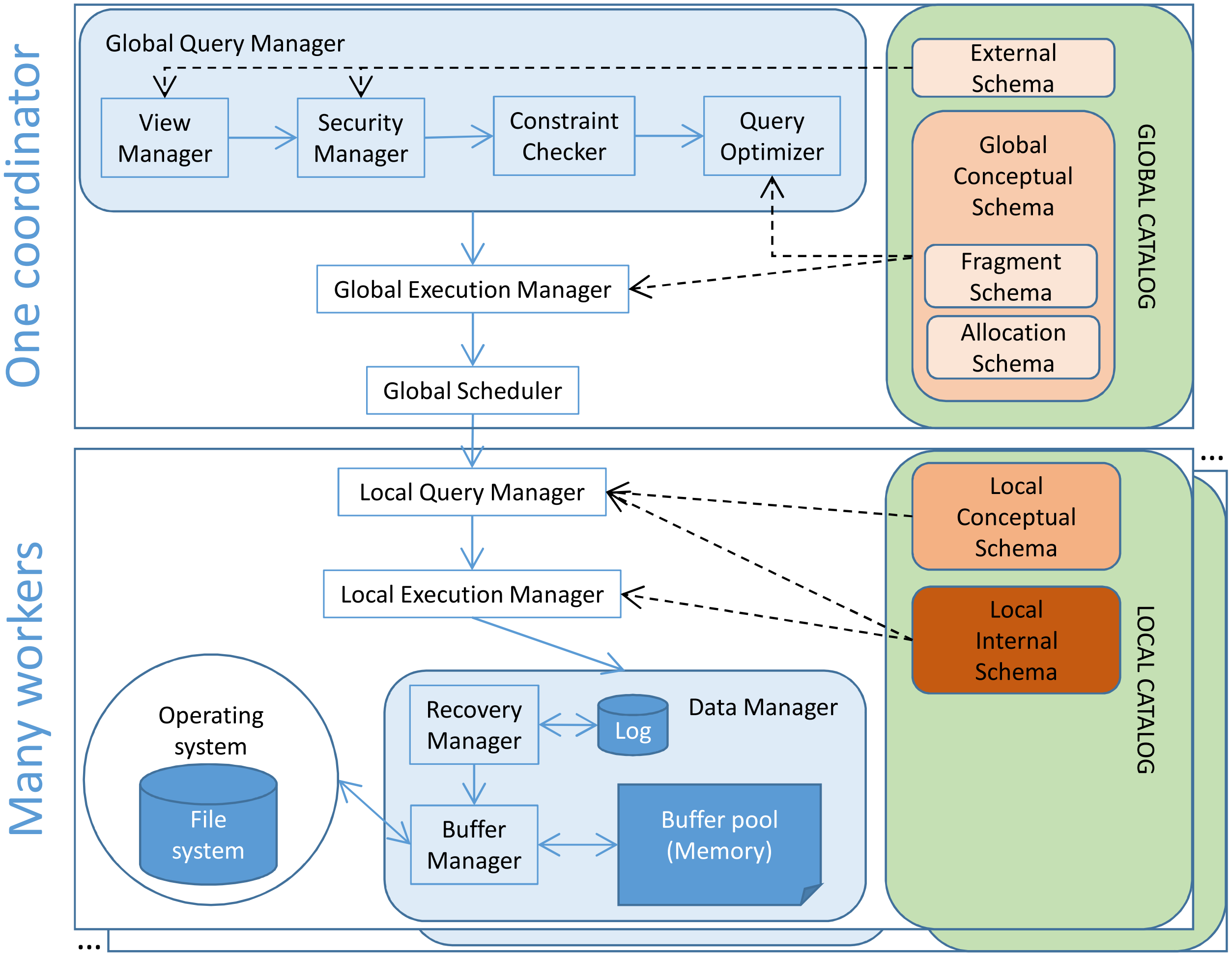


Querying

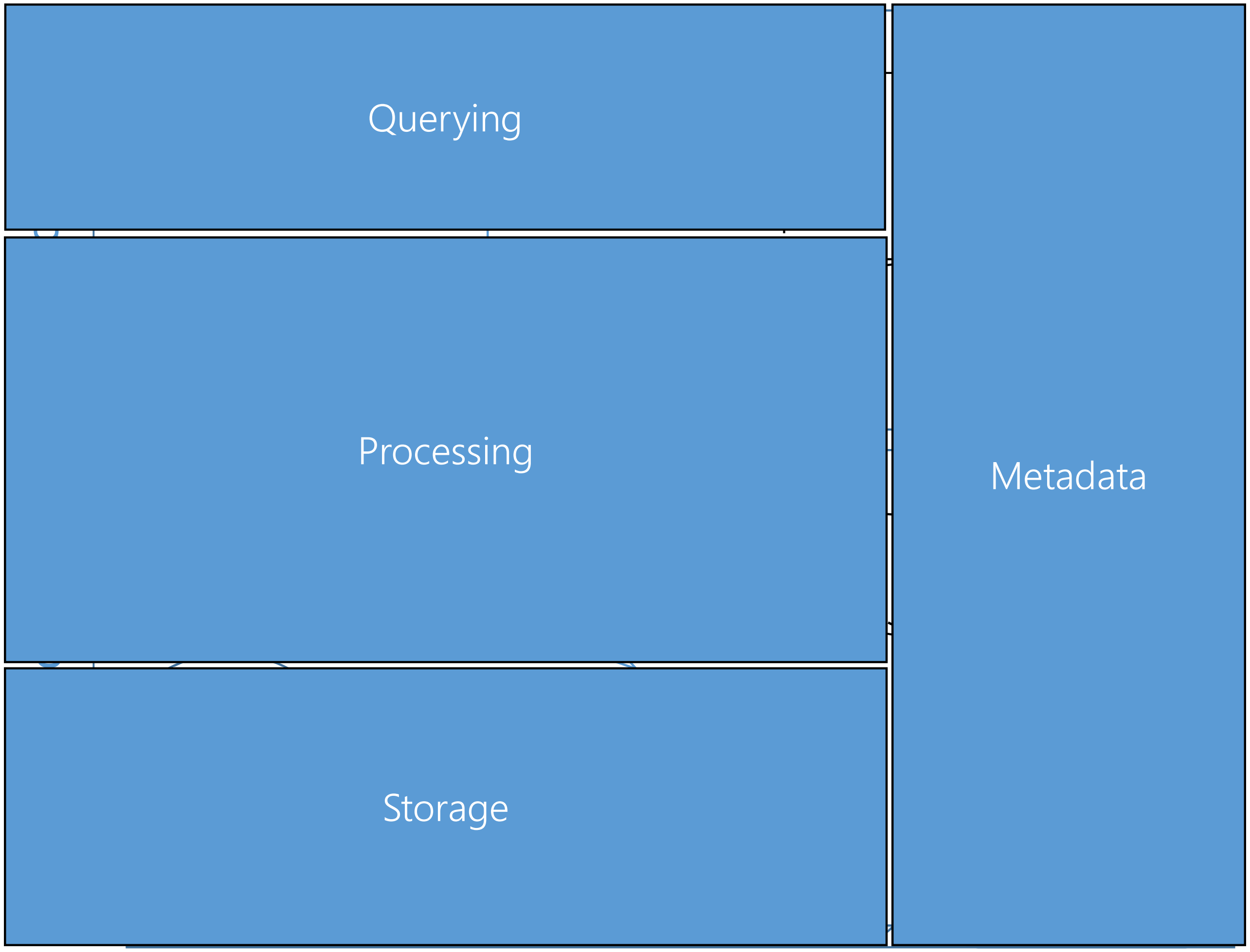
Processing

Storage

Distributed DBMS Architecture



Distributed DBMS Architecture



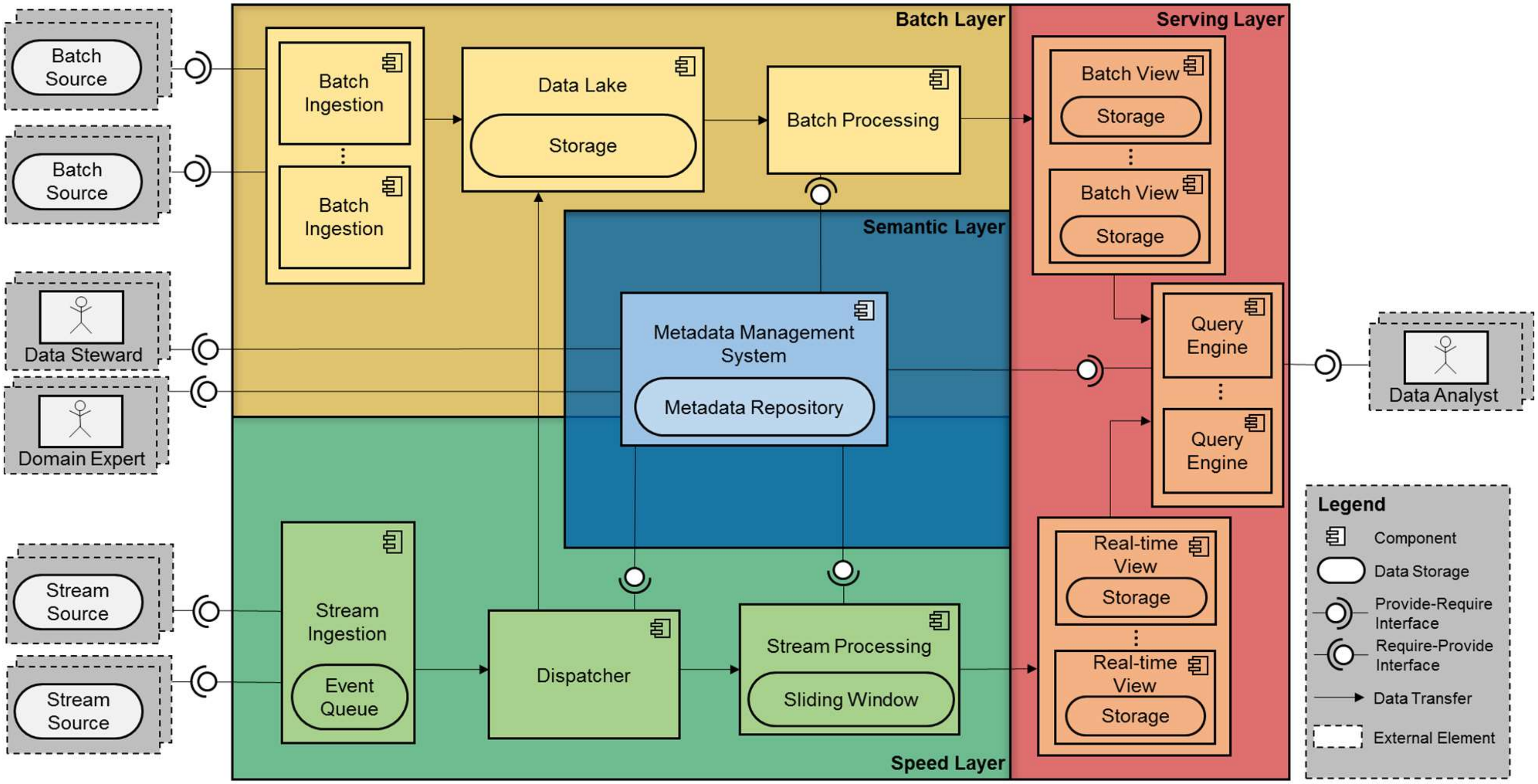
Querying

Processing

Storage

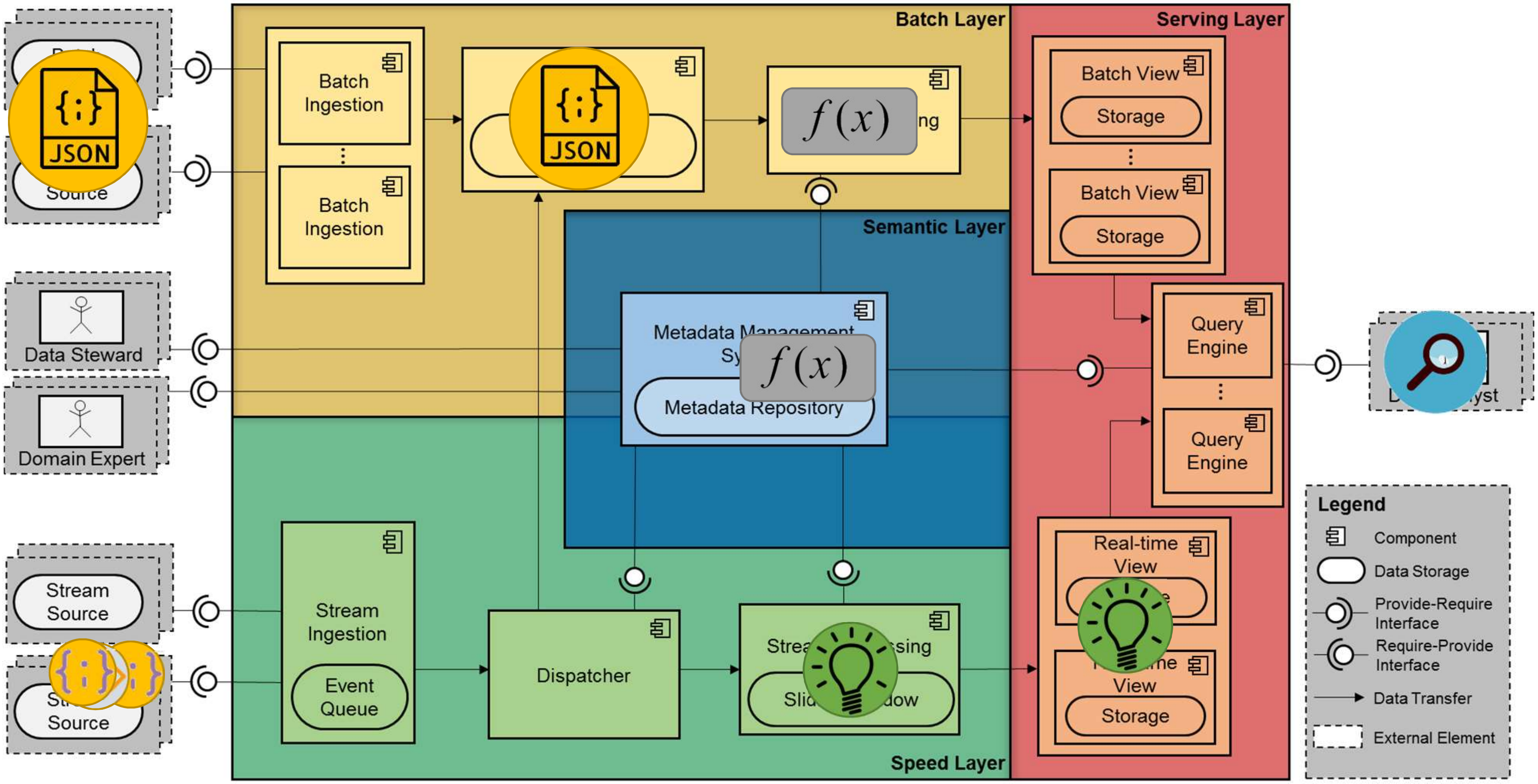
Metadata

Bolster

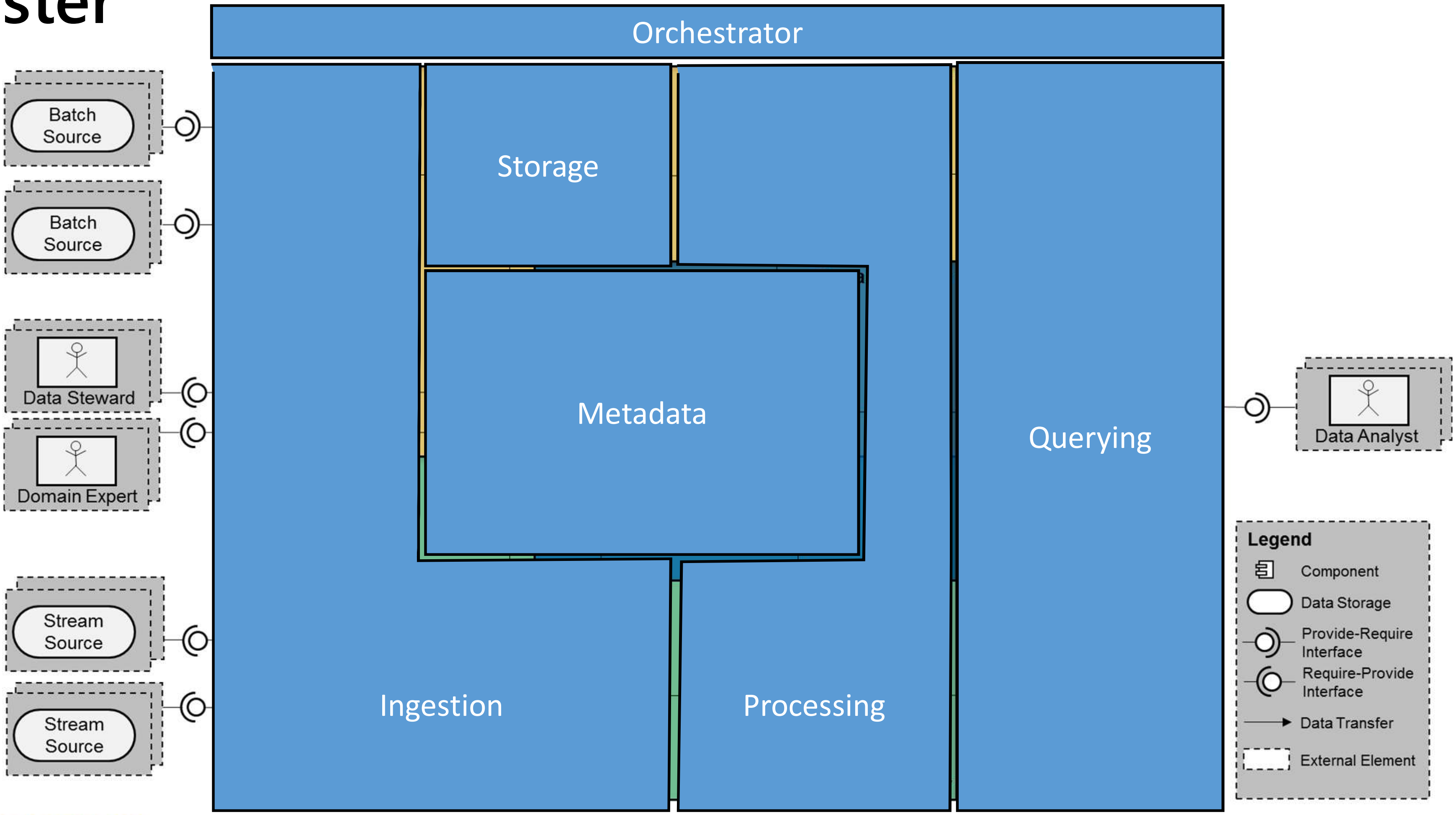


S. Nadal et al.

Bolster

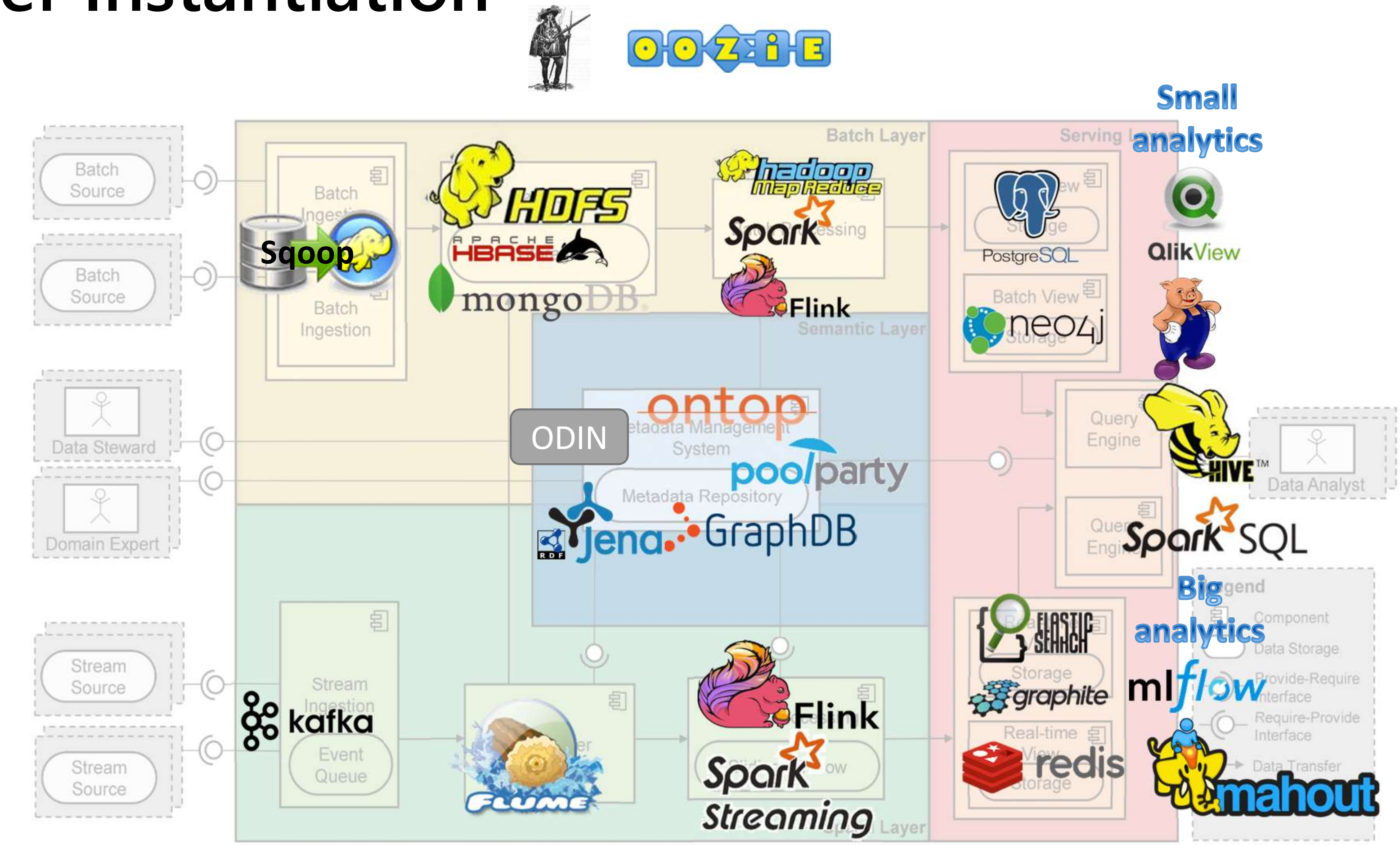

Batch Layer
Serving Layer
Semantic Layer
Speed Layer
Source
Data Steward
Domain Expert
Stream Source
Source
Batch Ingestion
Batch Ingestion
JSON
f(x)
ng
Batch View Storage
Batch View Storage
Metadata Management Sy
f(x)
Metadata Repository
Query Engine
Query Engine
D yst
Stream Ingestion
Event Queue
Dispatcher
Strea sing
Slid dow
Real-time View
View Storage
Legend
Component
Data Storage
Provide-Require Interface
Require-Provide Interface
Data Transfer
External Element

Bolster



S. Nadal et al.

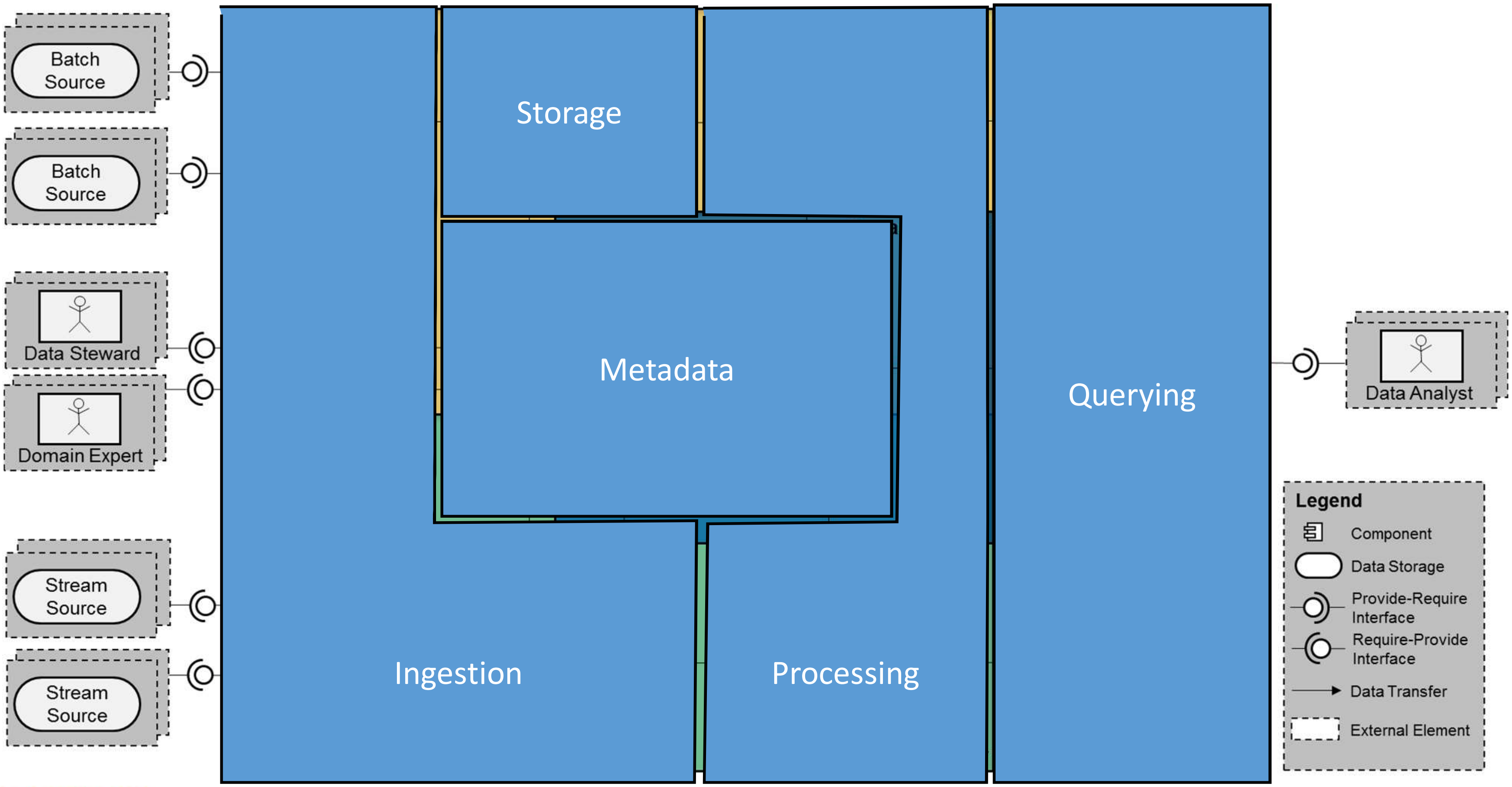
Bolster Instantiation



Big DataBase Management System in use

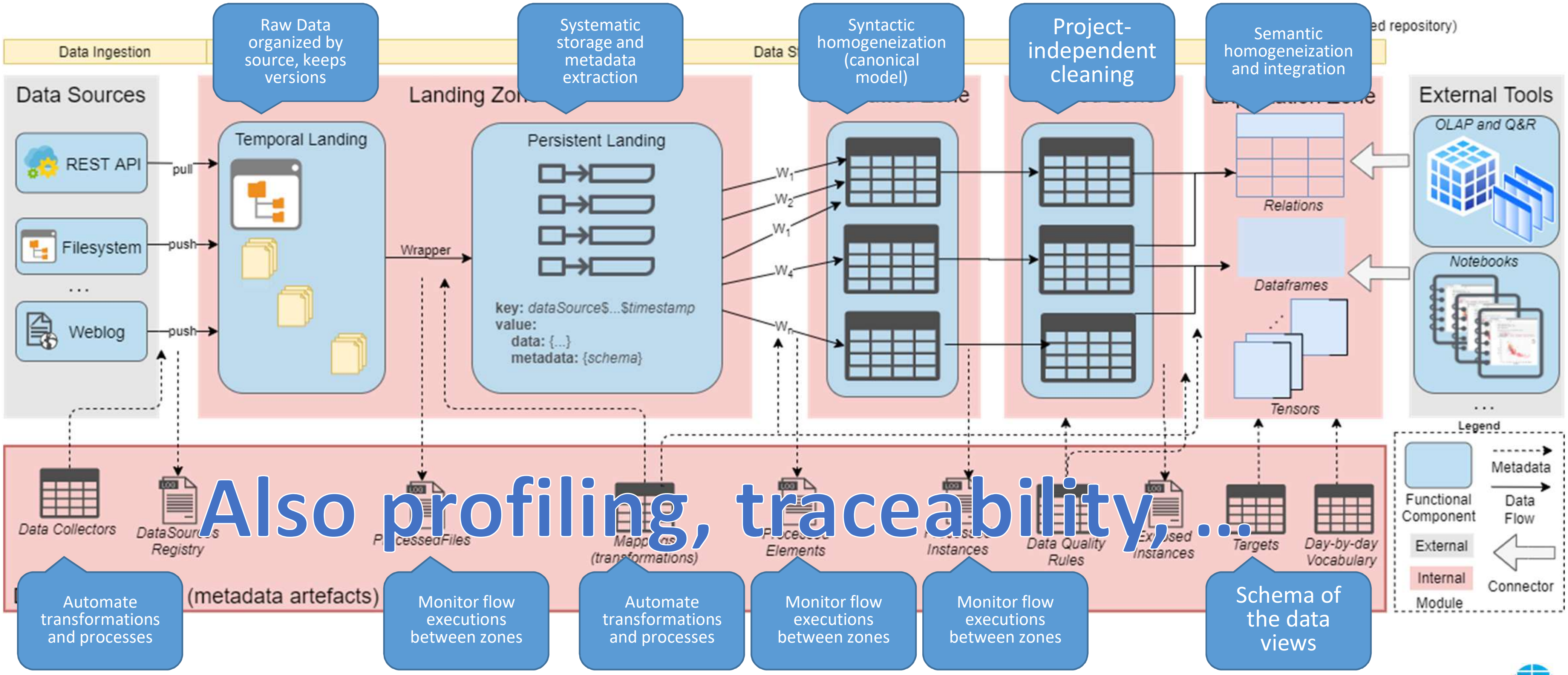
Configuration and data/metadata governance

Data Management



S. Nadal et al.

Data Management backbone



Raw Data organized by source, keeps versions

Systematic storage and metadata extraction

Syntactic homogeneization (canonical model)

Project-independent cleaning

Semantic homogeneization and integration

Also profiling, traceability, …

Automate transformations and processes

Monitor flow executions between zones

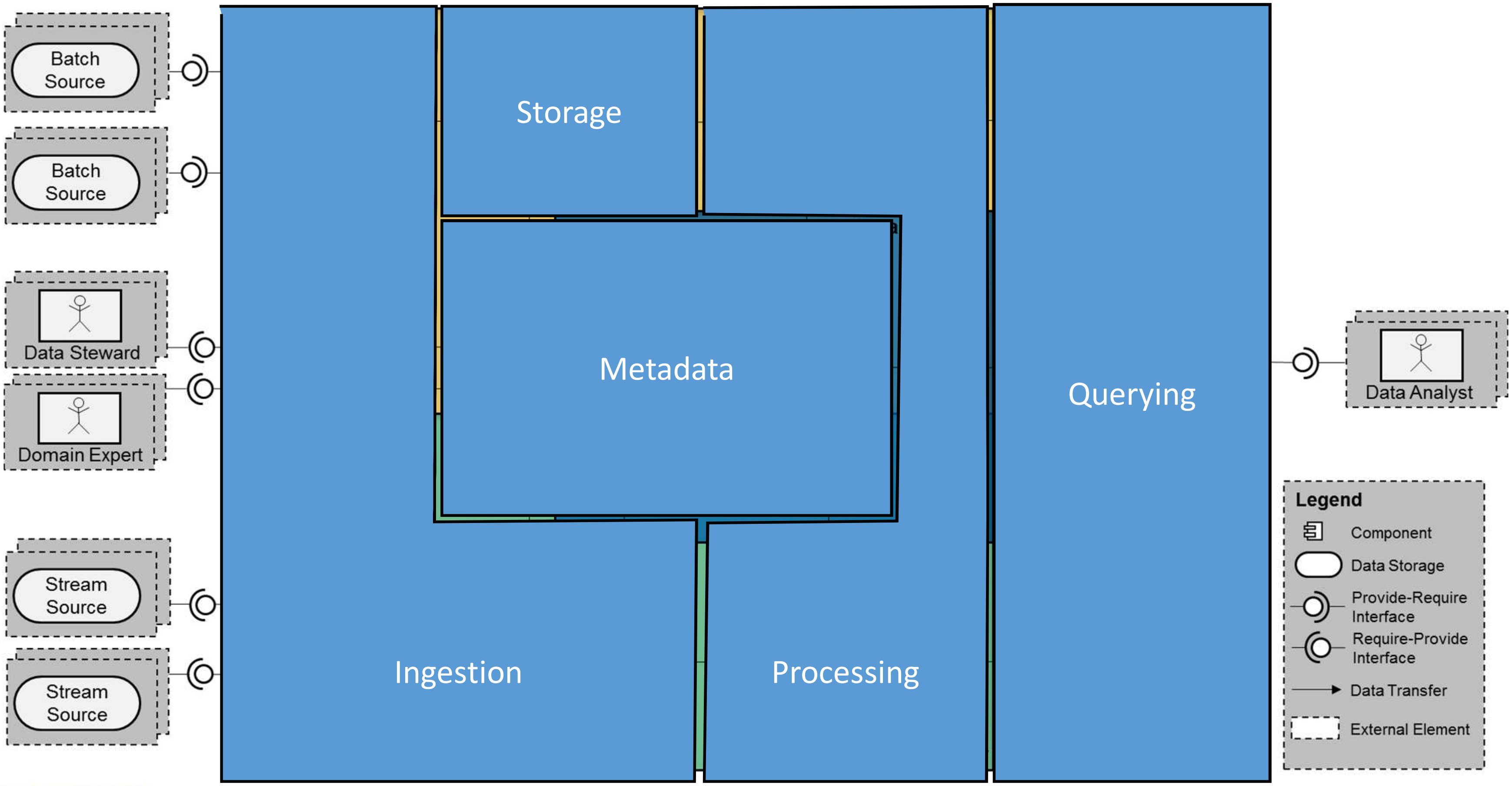
Automate transformations and processes

Monitor flow executions between zones

Monitor flow executions between zones

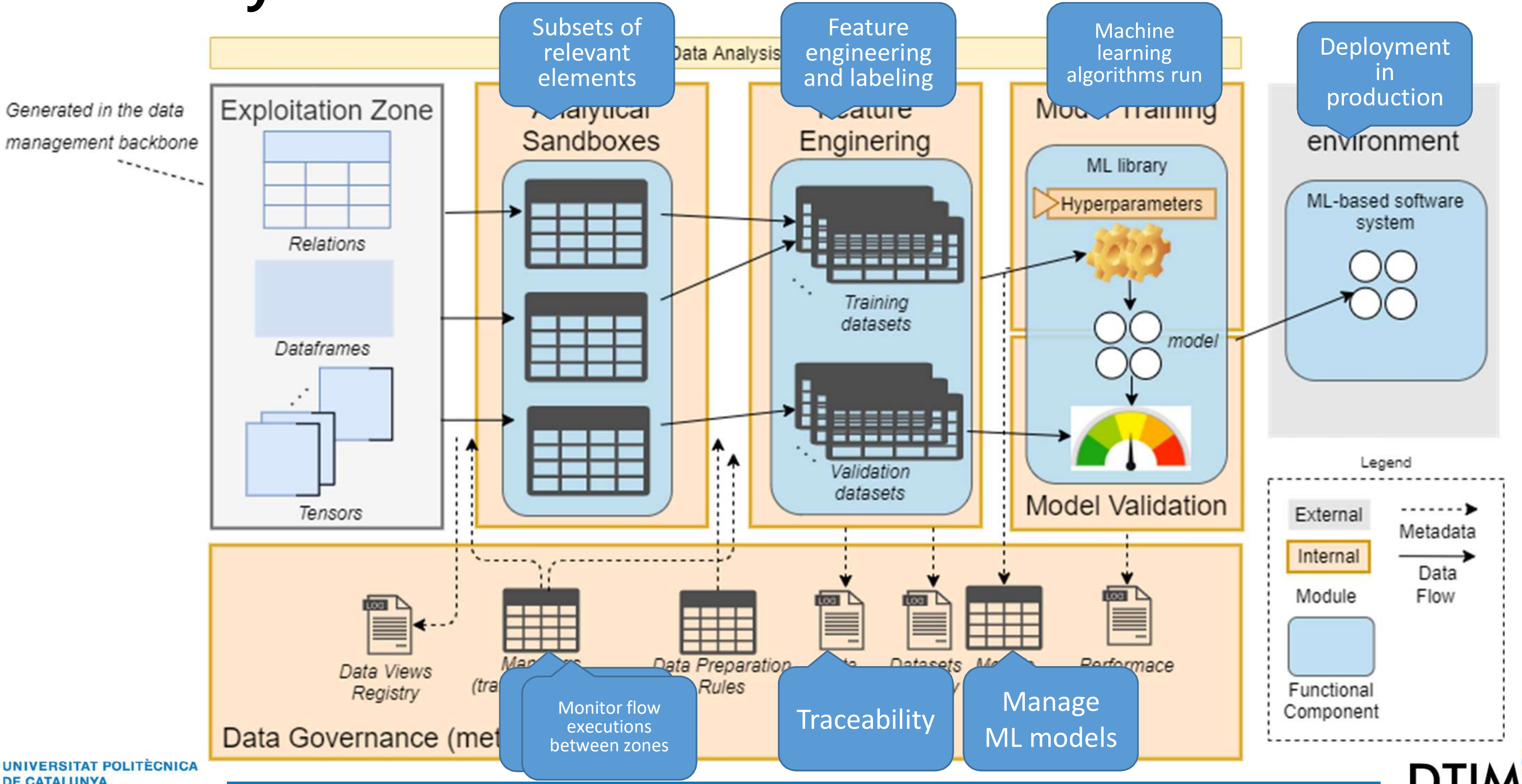
Schema of the data views

Data Analysis



S. Nadal et al.

Data Analysis Backbone



Subsets of relevant elements

Feature engineering and labeling

Machine learning algorithms run

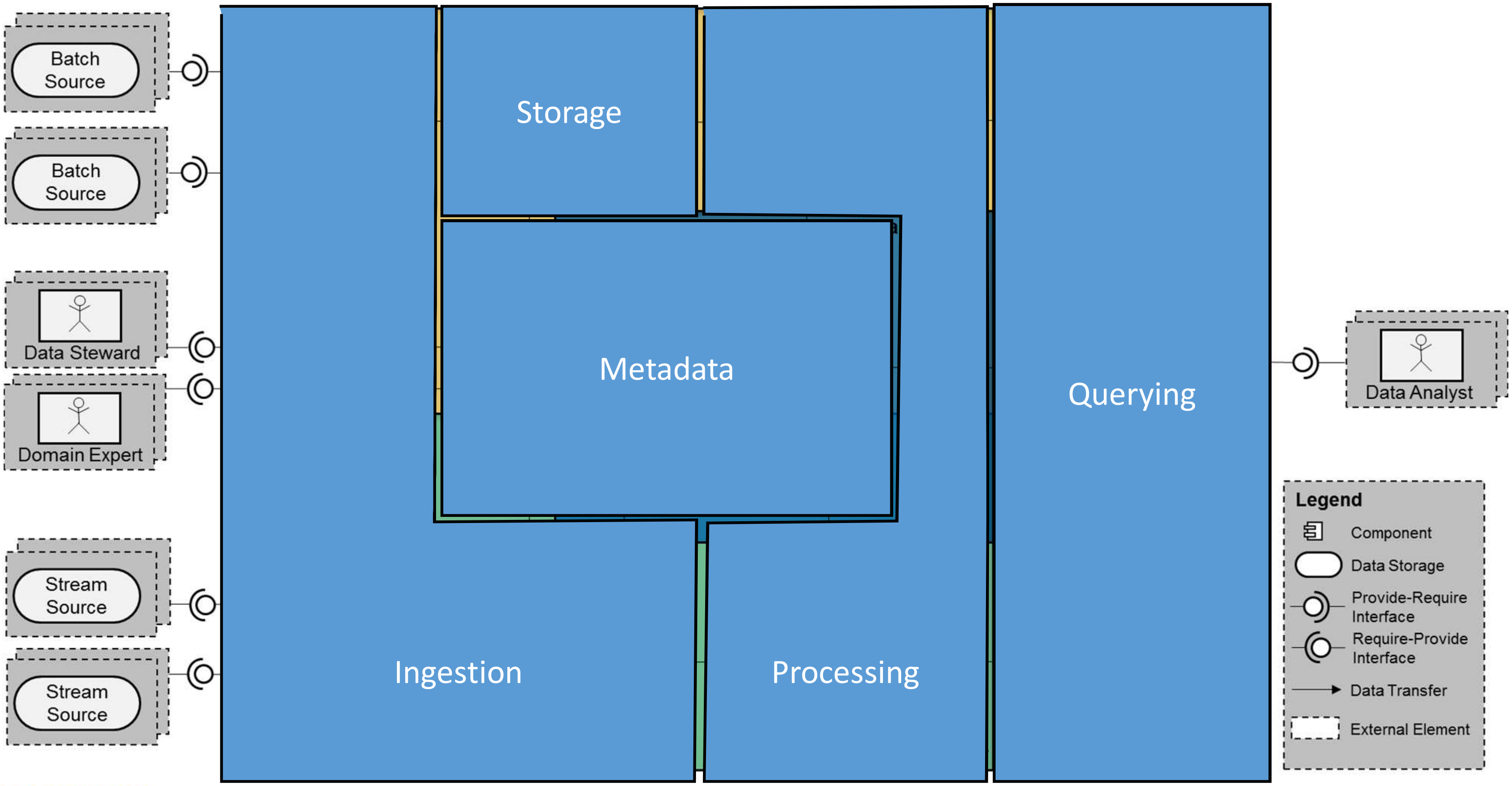
Deployment in production

Monitor flow executions between zones

Traceability

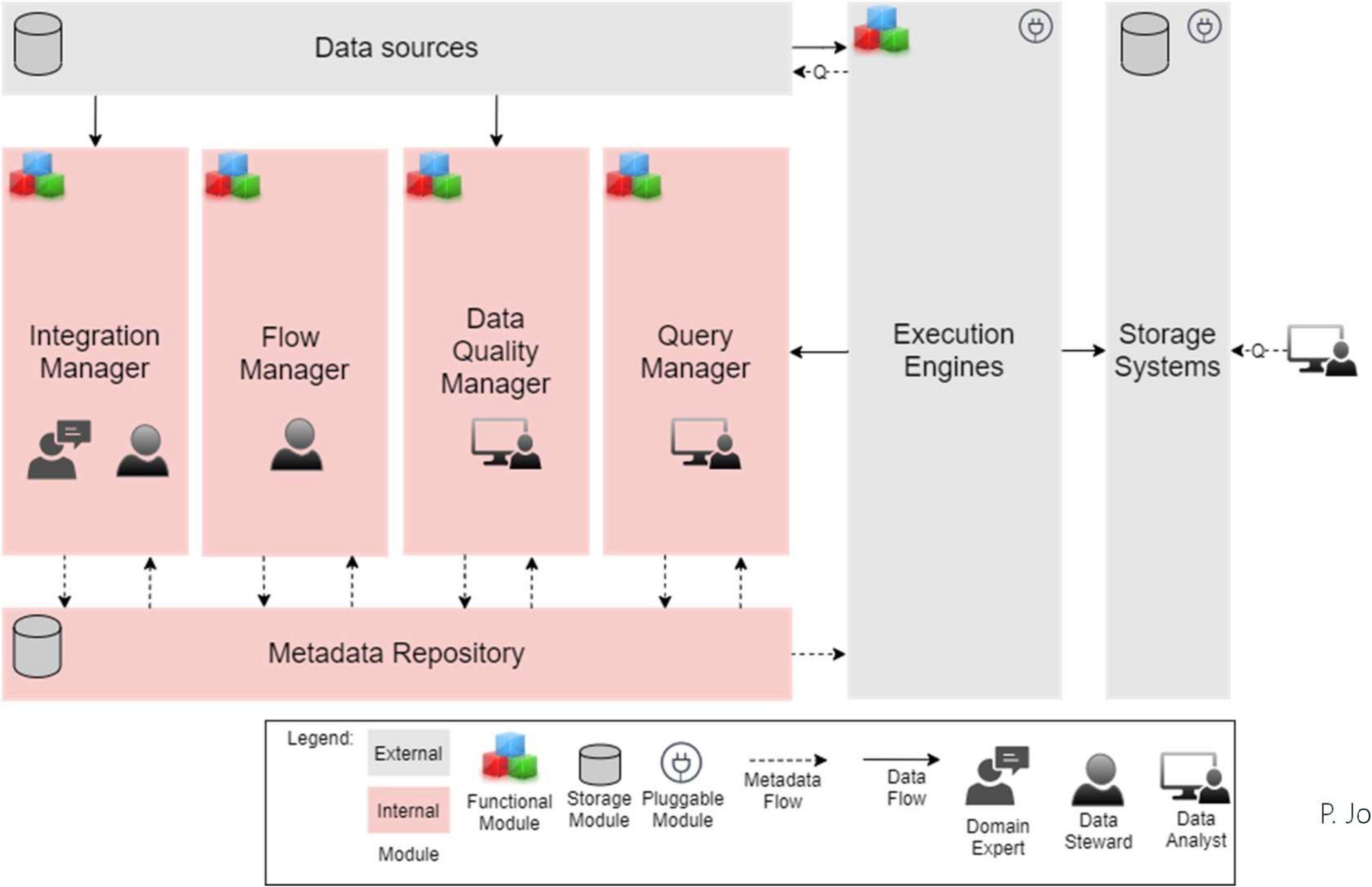
Manage ML models

Metadata Managent



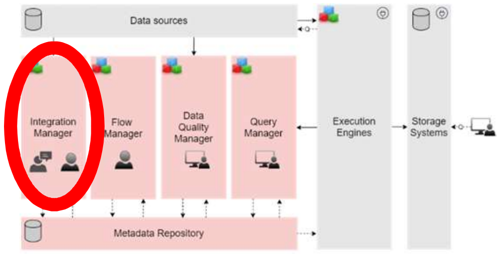
S. Nadal et al.

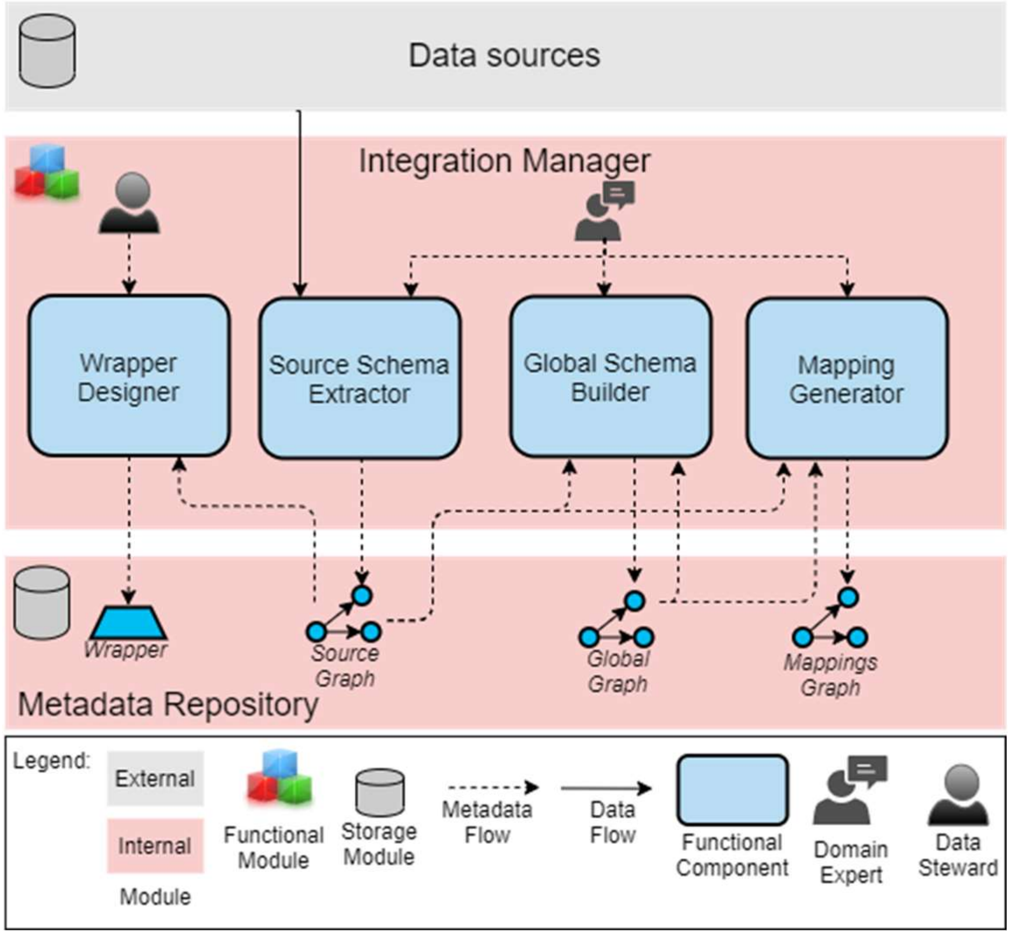
Metadata Manager



P. Jovanovic et al.

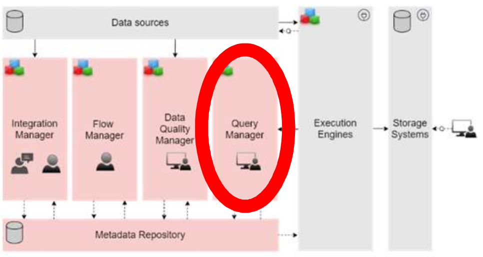
Metadata Manager (Integration)

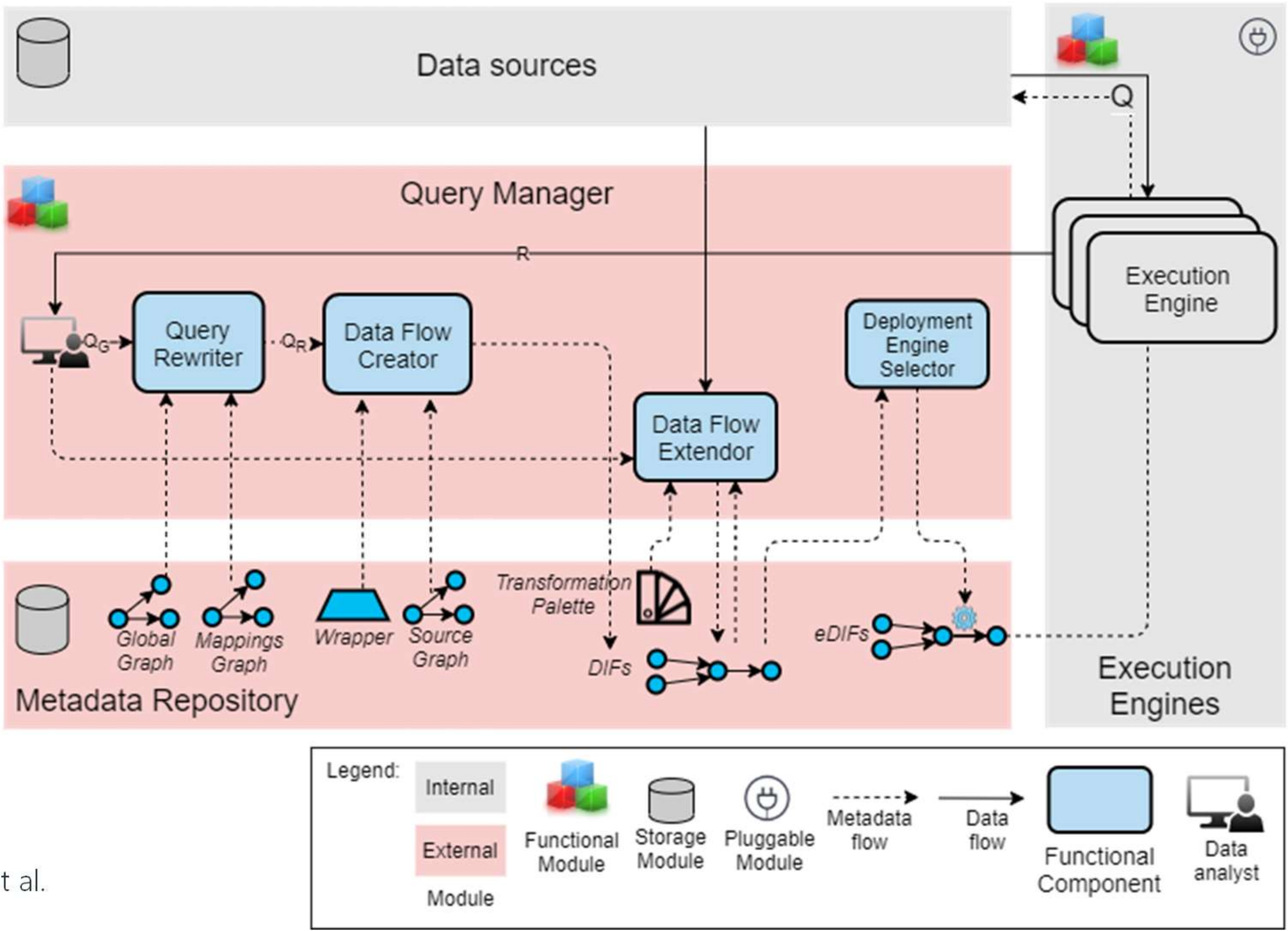




P. Jovanovic et al.

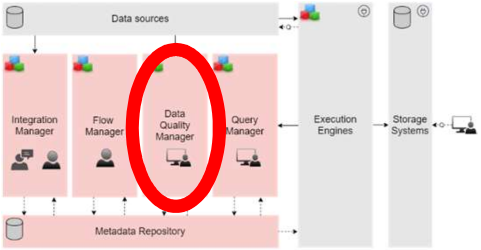
Metadata Manager (Querying)

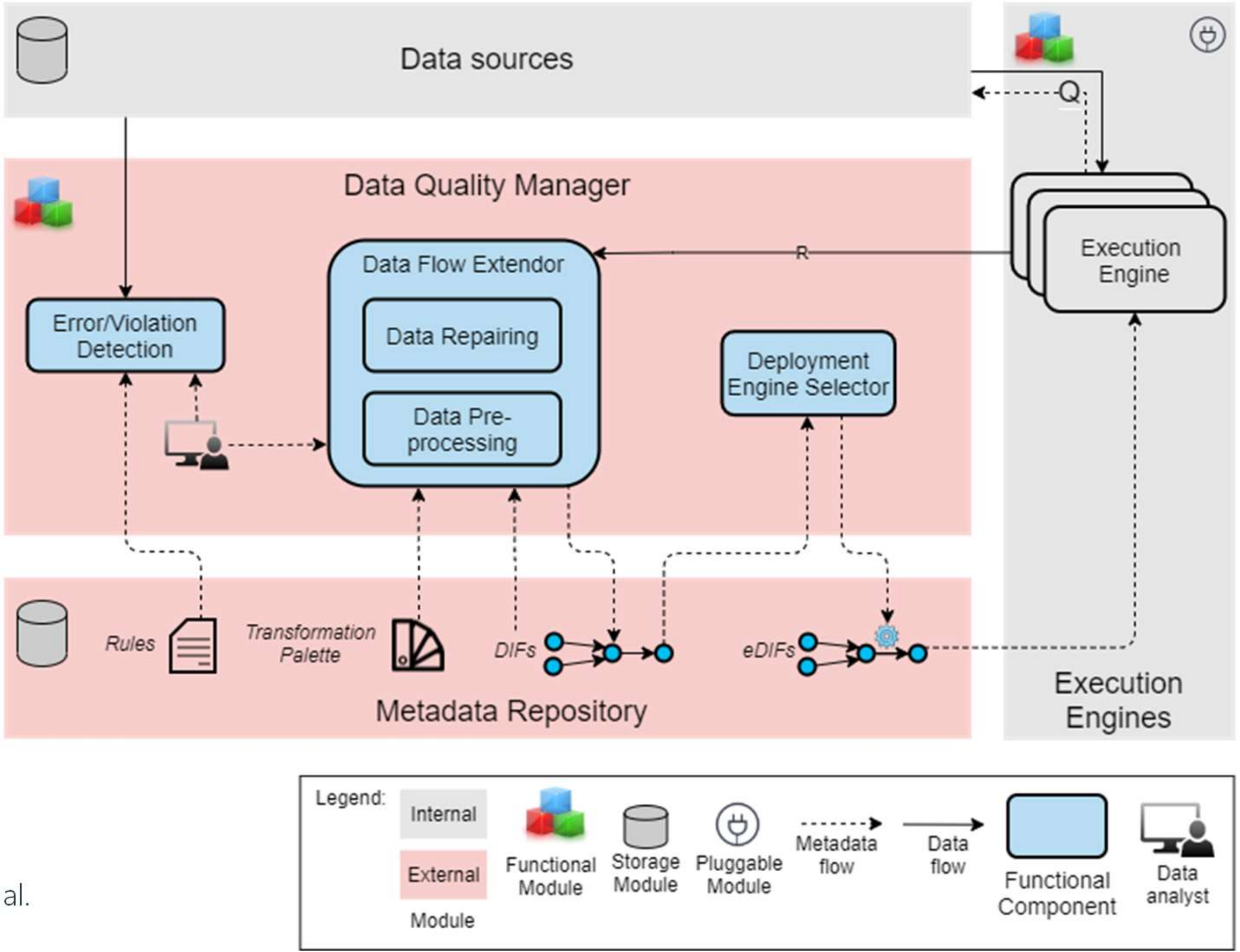




P. Jovanovic et al.

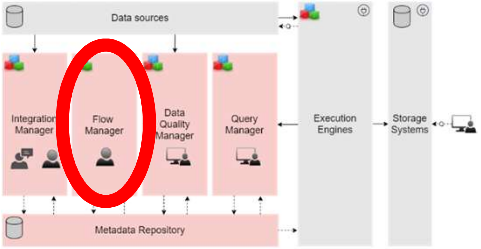
Metadata Manager (Data Quality)

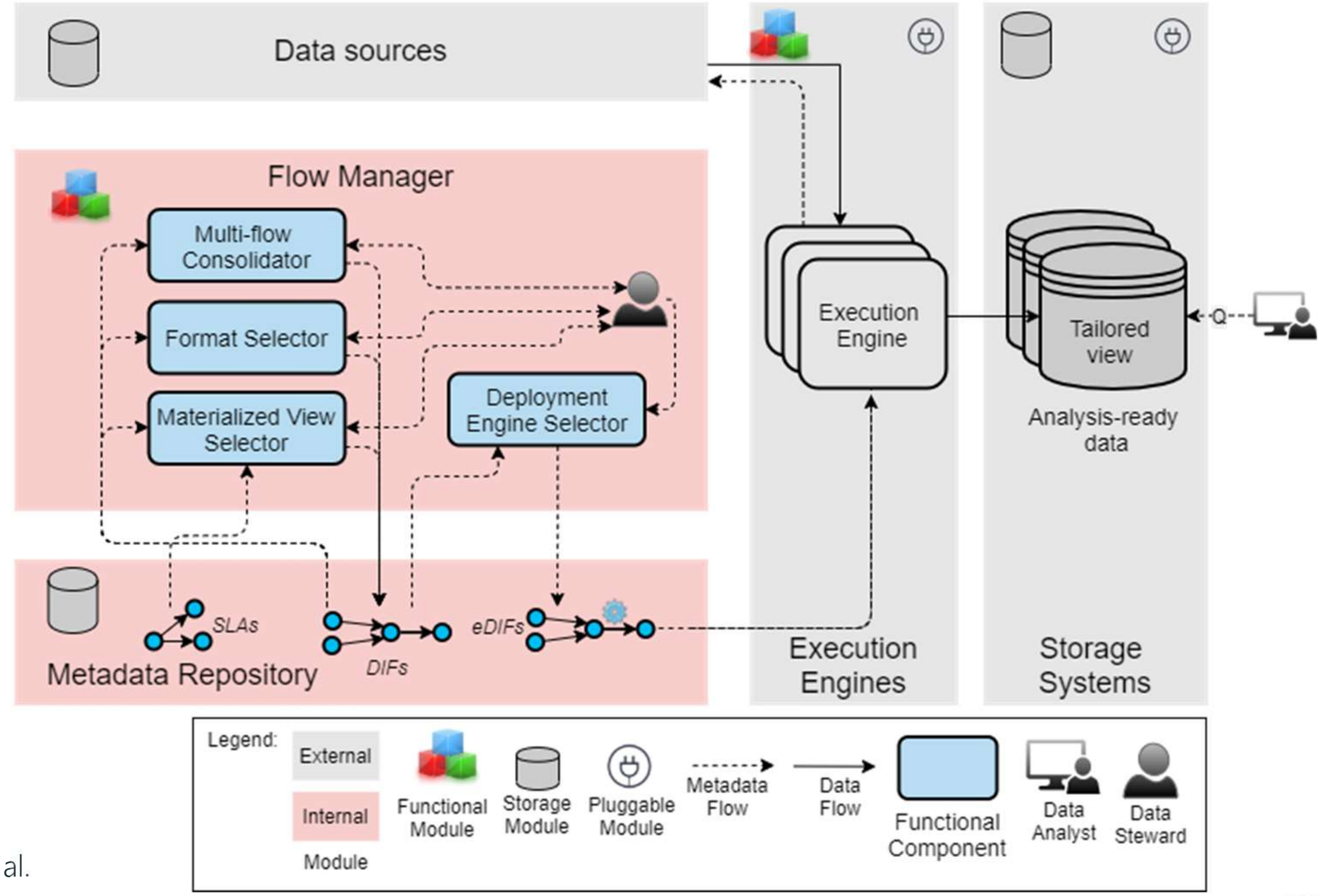




P. Jovanovic et al.

Metadata Manager (Data flows)





P. Jovanovic et al.

Closing

Conclusions

- Data engineering tasks in Big Data environments have some specificities
- New architectural solutions are needed
  - Data Lake
  - Lambda and Kappa
- There is a need for metadata
- Reference architectures have a clear correspondence with that of a DBMS
  - Bolster (to map components)
  - Quarry (to map metadata management and governance tools)

References

- S. Nadal et al. *A Software Reference Architecture for Semantic-Aware Big Data Systems*. Information and Software Technology 90. Elsevier, 2017
- P. Jovanovic et al. *Quarry: A User-centered Big Data Integration Platform*. Information Systems Frontier, 2021

- D. McCreary and A. Kelly. *Making Sense of NoSQL*. Manning, 2014
- M. Grover et al. *Hadoop Application Architectures*. O'Reilly, 2015
- N. Marz and J. Warren. *Big Data: Principles and best practices of scalable realtime data systems*. Manning Publications Co., 2015
- D. Sculley te al. Hidden technical debt in machine learning systems. In Advances in Neural Information Processing Systems, 28. Curran Associates, Inc., 2015.
- S. Nadal et al. *ODIN: A Dataspace Management System*. International Semantic Web Conference, 2019