# Architectural Approach to Metadata-Driven ETL Frameworks

*By Dmitriy Lychev*

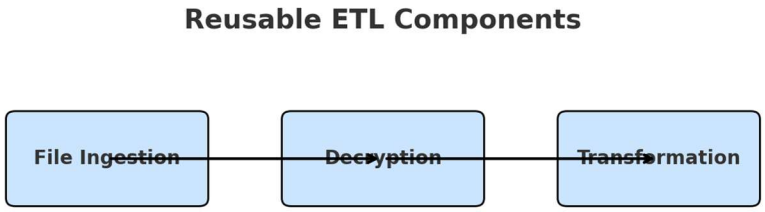
Reusable, Scalable, and Cloud-Ready ETL Design (SSIS, Talend, Airflow via Docker)

## 1. Introduction

This document outlines an architectural approach to building metadata-driven ETL frameworks. The solution was successfully implemented using Microsoft SSIS, Talend, and Apache Airflow running containerized in Docker. The goal is to ensure reusability, scalability, and cloud readiness while maintaining robust auditing, logging, and orchestration across heterogeneous platforms.

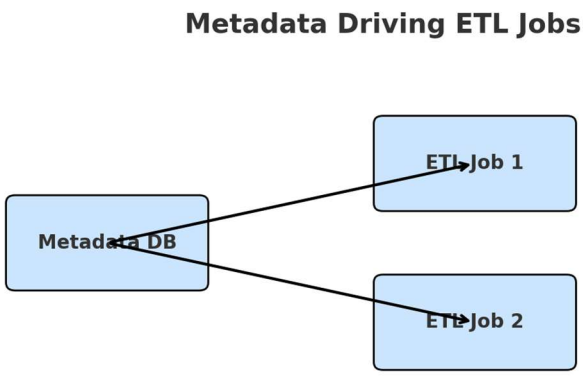
## 2. Reusable Components

Reusable components are the foundation of the framework. These are pre-built modules for common tasks like SFTP file ingestion, decryption, validation, and data transformation. In SSIS, this was achieved with parameterized packages; in Talend, with reusable joblets; and in Airflow, with Python operators running in Docker containers.



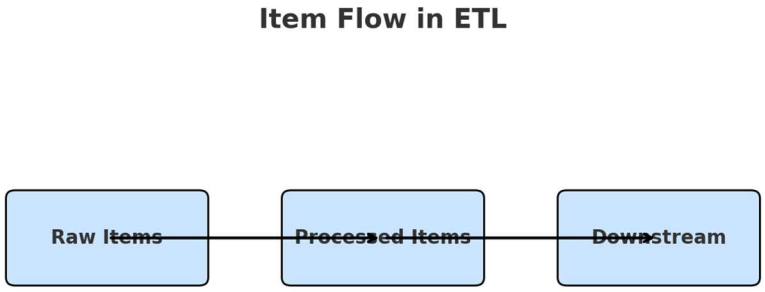

## 3. Metadata Management

All workflow logic and runtime parameters are driven by metadata stored in a central repository (MS SQL Server or MySQL). This includes job definitions, schedule information, and audit logs. Two levels of logging are captured: workflow-level and step-level. This allows precise monitoring and troubleshooting across SSIS, Talend, and Airflow implementations.




## 4. Item Management System

The Item Management System tracks which files or data sets have been processed. It prevents duplicate ingestion and ensures completeness by recording attributes like file name, size, and timestamp.




## 5. Boundary Management

Boundary management ensures only the correct range of data is processed. This is essential for incremental loads. For example:

- In SSIS, boundaries were stored in control tables and used in WHERE clauses.
- In Talend, context variables read boundary values dynamically.
- In Airflow, DAG parameters and Jinja templating controlled the extract windows.

SQL Example (using timestamp boundary):

```
SELECT * FROM orders WHERE order_date > @last_boundary AND
order_date <= GETDATE();
```

This ensures only new or updated records are pulled since the last successful execution.

[Diagram: Boundary Concept with Rolling Window Extract]

## 6. Dynamic Properties

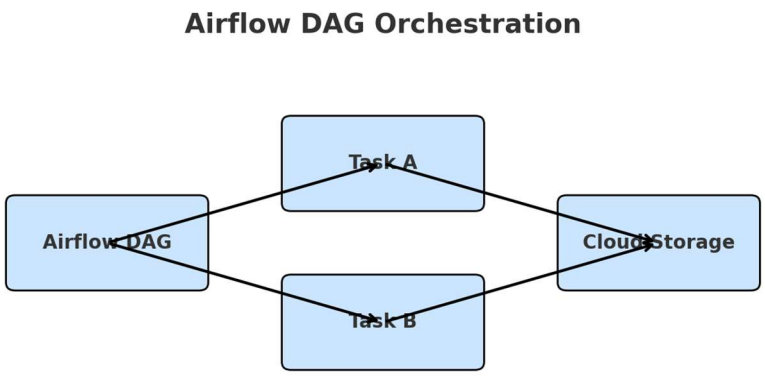
Dynamic properties enable flexibility across different environments and partners. Instead of hardcoding file paths or names, values are generated at runtime. For example, a filename pattern like 'file_to_partner_|||date|||.csv' is dynamically resolved into 'file_to_partner_2025_09_15.csv'.

```
SQL Function Example:
SELECT generate_dynamic_string_universal_tz(
  's3://bucket/partner/|||date|||/orders.csv',
  '%Y_%m_%d',
  'US/Eastern'
);
```

This dynamic property generation was leveraged in Talend (contexts), SSIS (configuration tables), and Airflow (Python functions with Jinja templates inside Dockerized tasks).

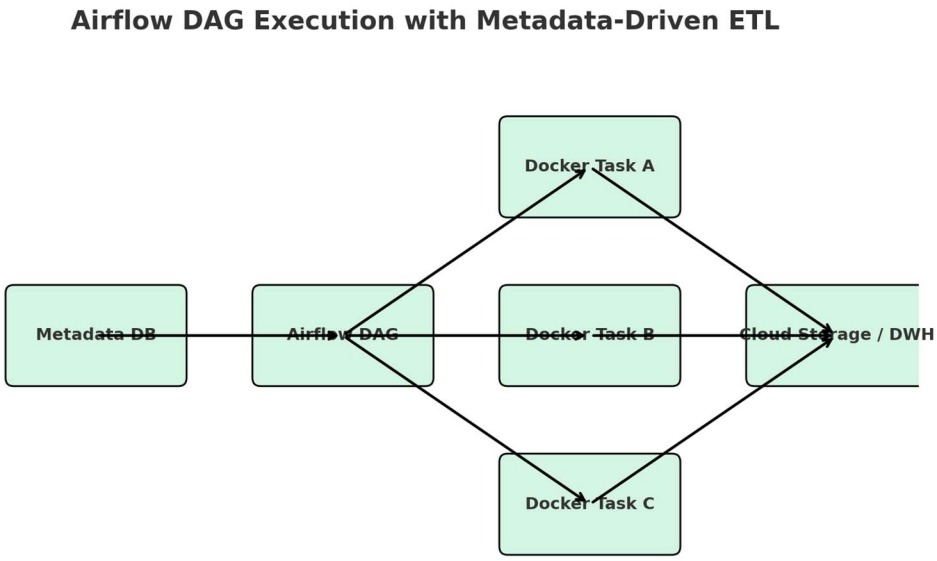
## 7. Orchestration Example (Airflow DAG)

Apache Airflow orchestrates ETL pipelines using metadata-driven DAGs. Each DAG dynamically determines which tasks to execute based on the metadata repository. Containerized Python ETL tasks run as Docker images, ensuring portability and scalability.



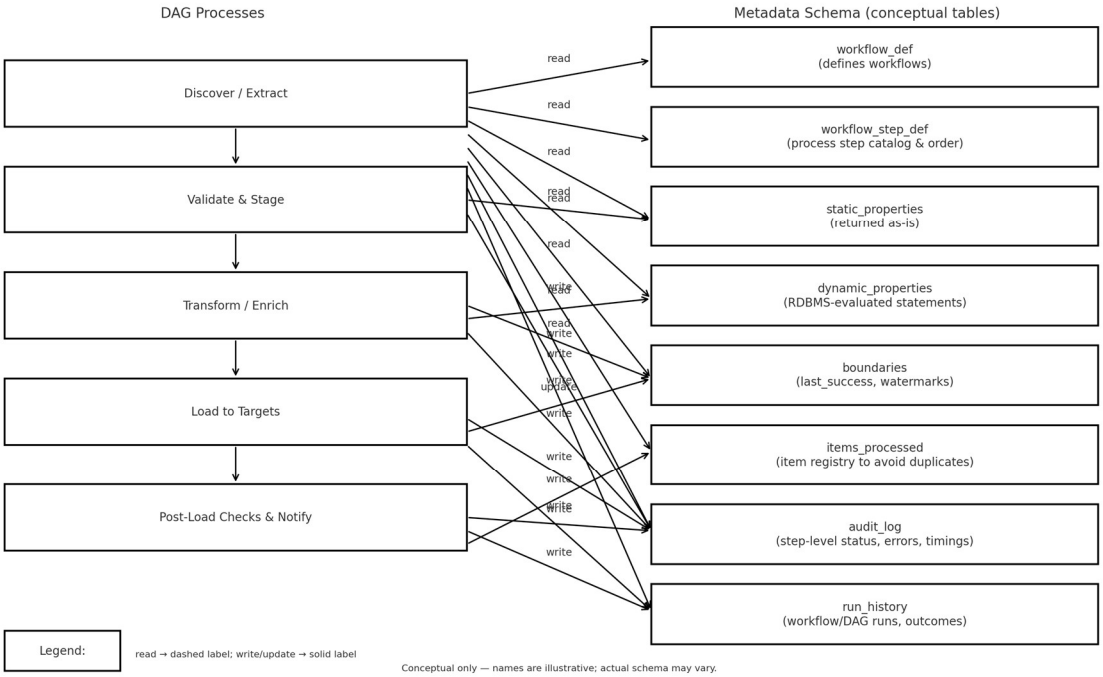

## 8. DAG Execution Example

The diagram below illustrates how metadata feeds Airflow DAG execution. The DAG triggers containerized tasks (Docker images) in parallel, each performing specific ETL steps, and loading results into Cloud Storage or a Data Warehouse.



## 9 Metadata Map

The diagram shows how DAG processes consult these tables alongside workflow definitions, boundaries, and audit/run tables to drive execution and capture lineage.




## 10. Conclusion

This architectural approach ensures modular, metadata-driven ETL adaptable across SSIS, Talend, and Airflow. It provides scalability, reusability, and auditability, while supporting seamless migrations to cloud-native platforms like Google BigQuery and AWS S3. With reusable components, dynamic properties, and robust boundary management, the framework reduces operational risk and accelerates development.