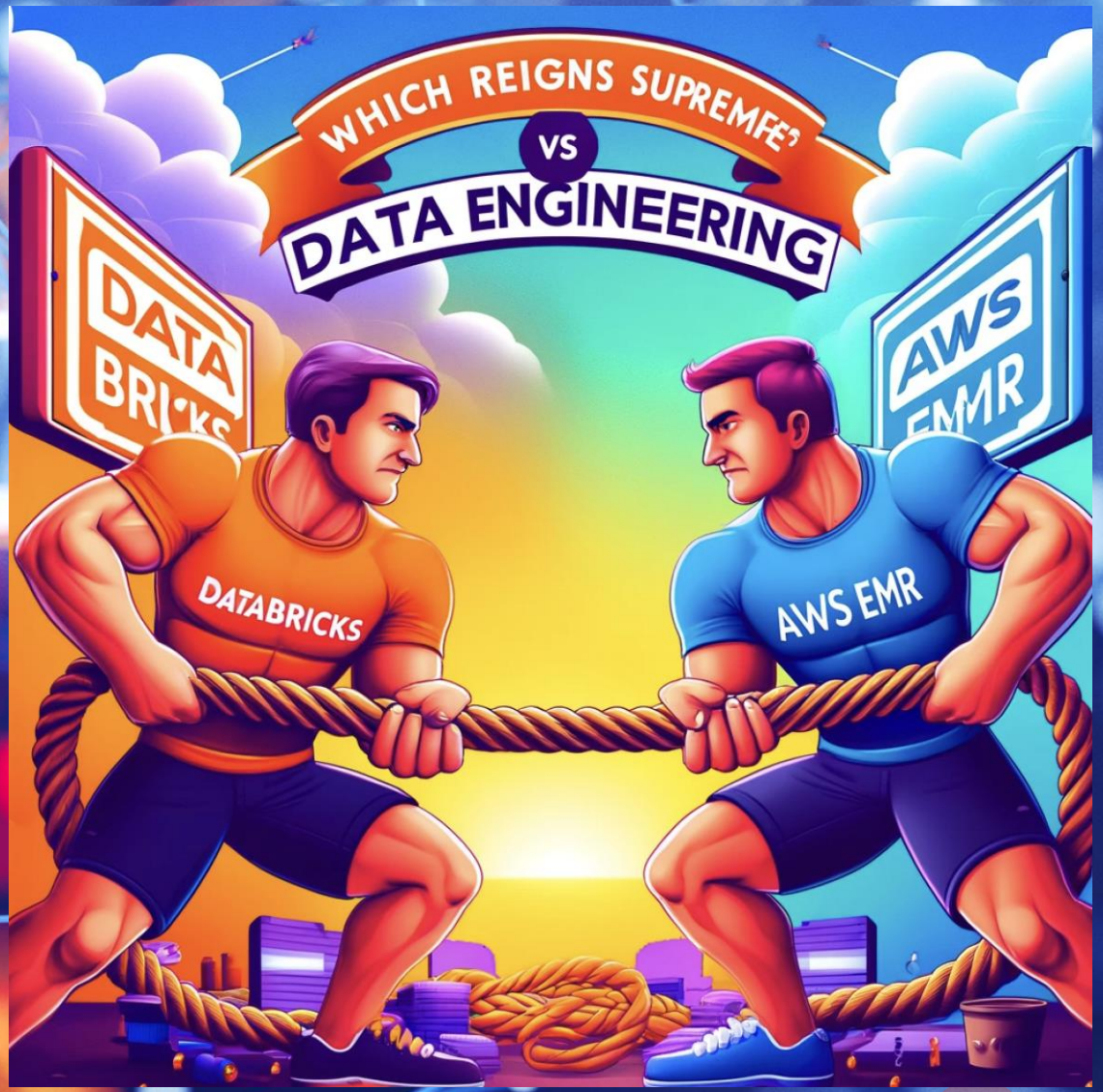
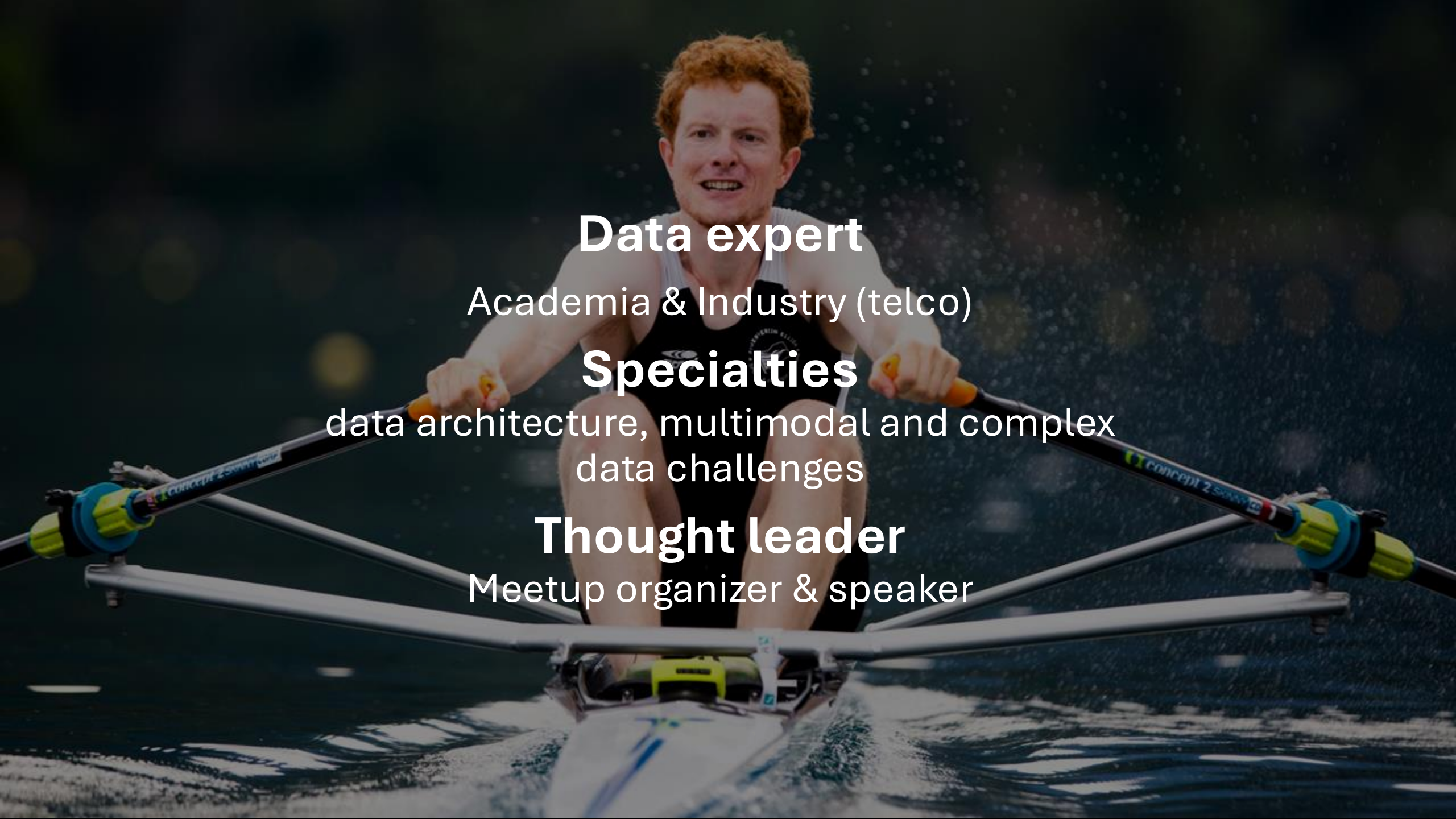
COST EFFICIENT ALTERNATIVE TO DATABRICKS

Georg Heiler



Exploring Alternatives for Cost-Effective and Flexible Data Pipelines → bit.ly/efficient-spark



Data expert

Academia & Industry (telco)

Specialties

data architecture, multimodal and complex data challenges

Thought leader

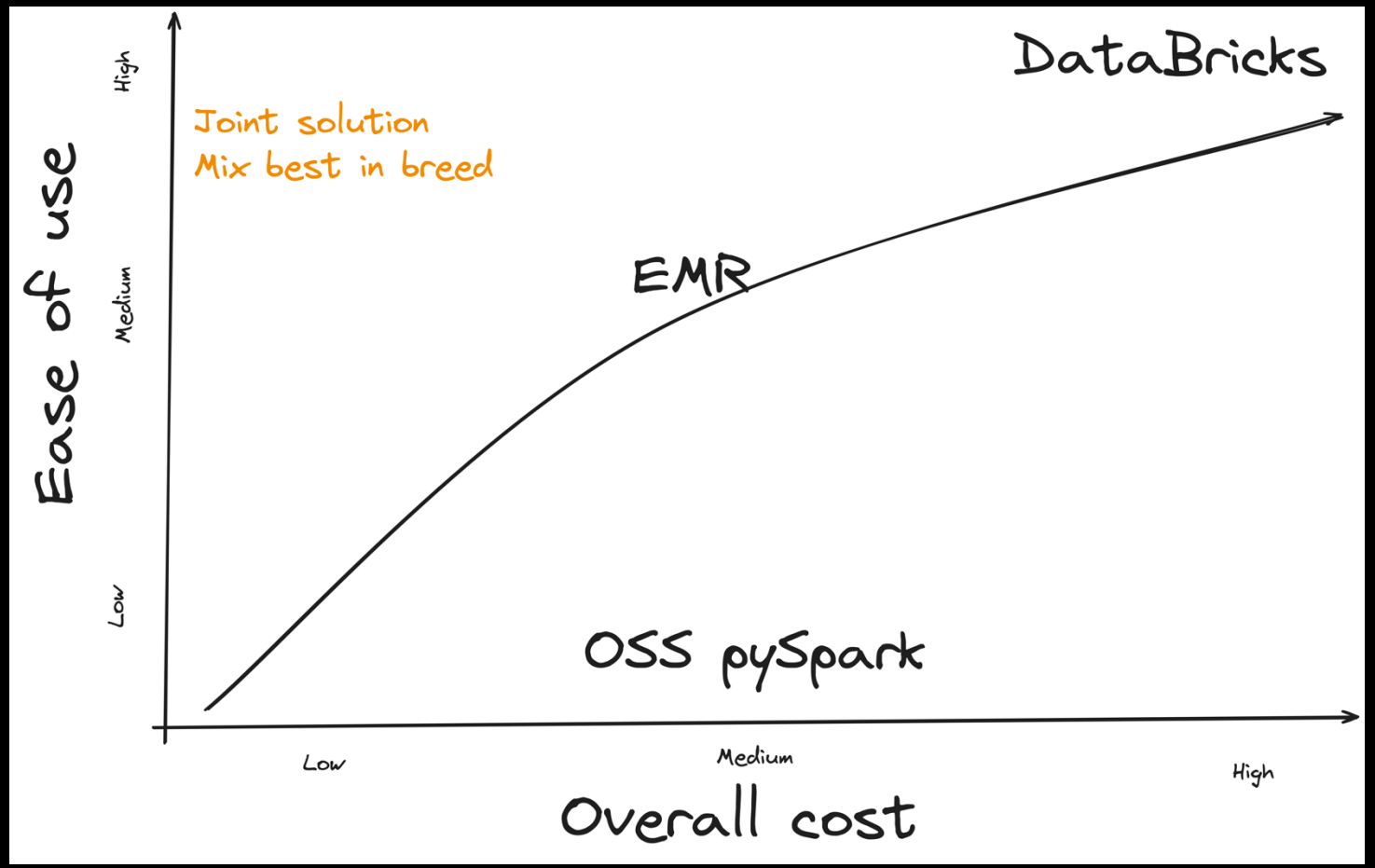
Meetup organizer & speaker

ASCI! Supply Chain Intelligence Institute Austria

- Rising importance of understanding and shaping supply chains (covid, Ukraine war)
- No fine-grained clean data accessible
- Abundant un- and semistructured data → sophisticated cleaning & parsing required
- Extract and classify links based on semantic context

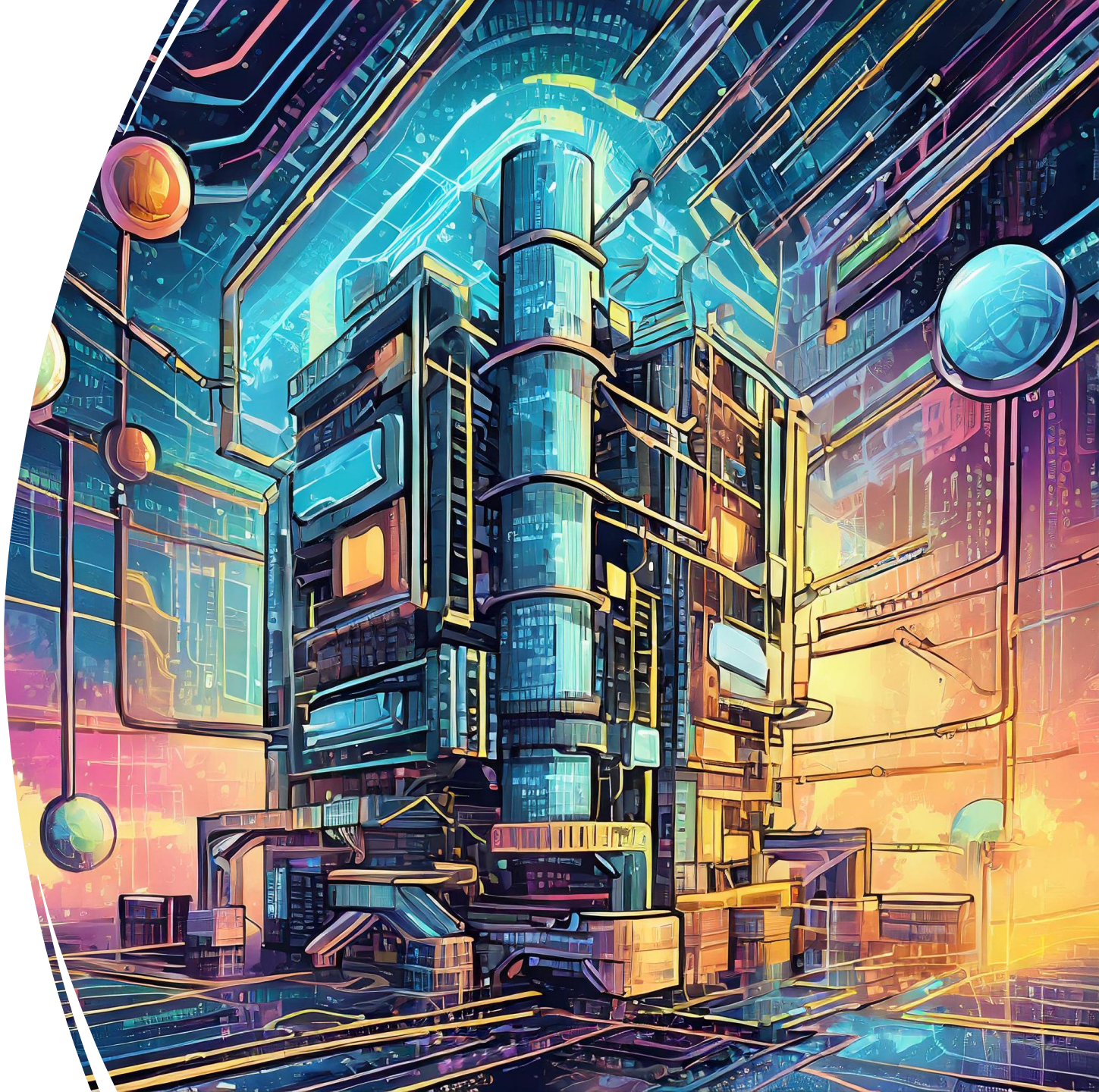
Results at a glance

- 43% Cost Reduction
- Software Engineering practices
- Future proof flexibility
- Single pane of glass for pipelines



History

- Mainframe
- Data warehouse
- Big Data (Hadoop)
- SQL on large data (Hive, Spark)
- Cloud DWH (Snowflake, bigquery)



PaaS offering

PaaS Solution Comparison

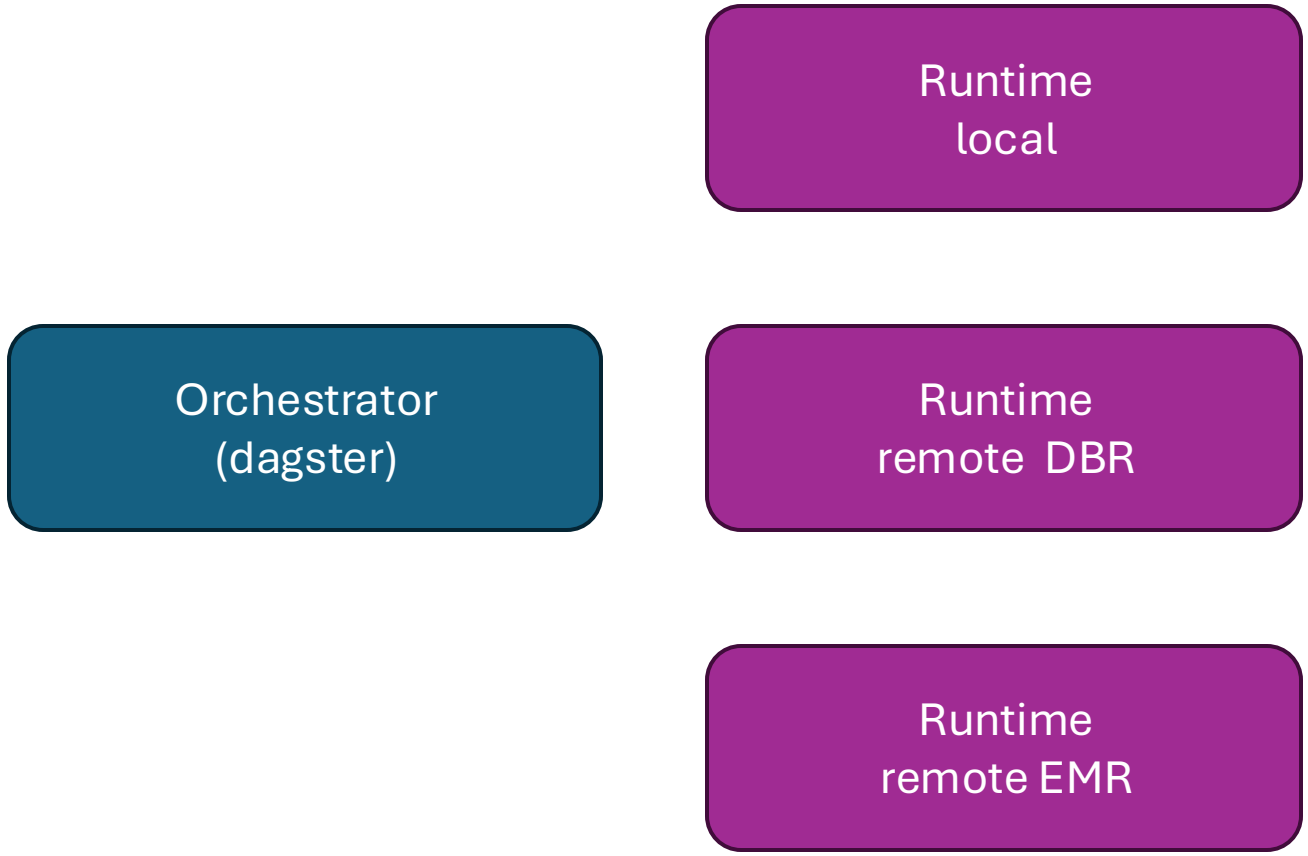
Databricks (DBR)	AWS Elastic Map Reduce (EMR)
• Easy to use • Can be expensive • Lock-in features (permissions, catalog) • Proprietary Photon engine	• Price efficient • Many tuning knobs available (& required) • OSS Spark managed (scaled)

Challenges

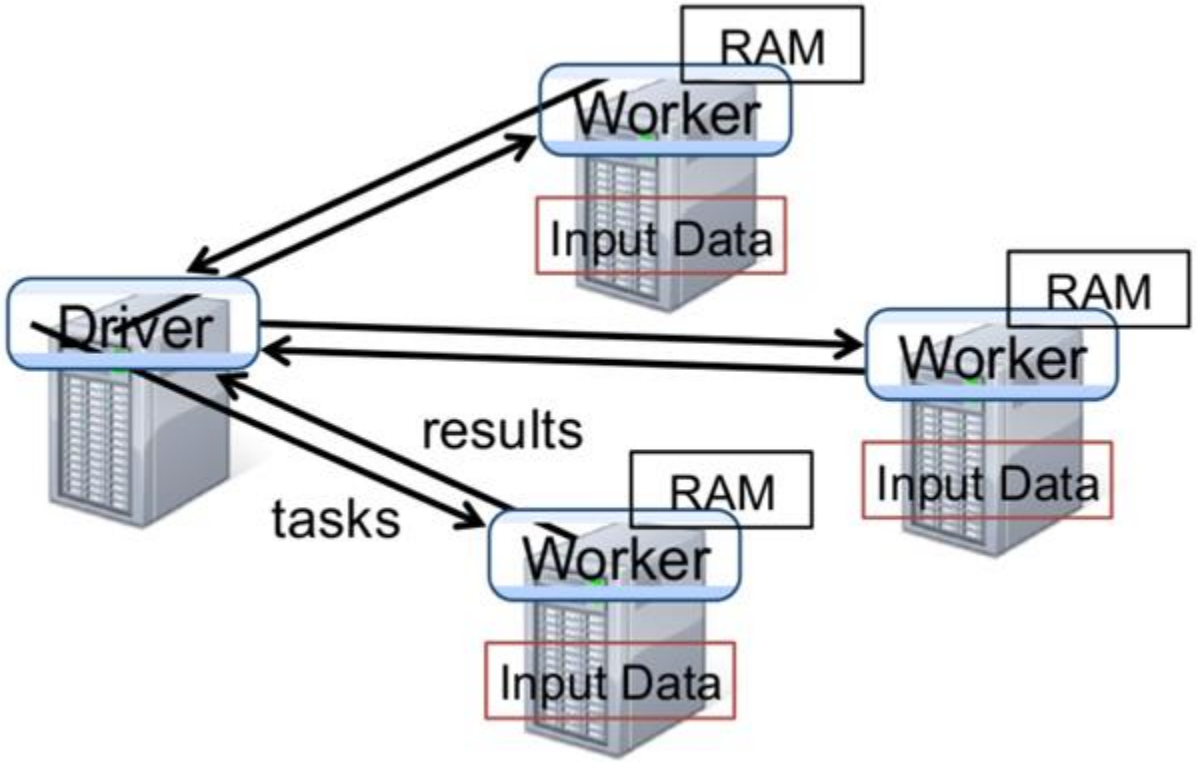
- Runaway expenses (usage-based pricing)
- Missing software engineering best practices (notebooks)
- Developer productivity reduced
- Vendor lock-in

Vision

- 0-cost switch
- Software engineering practices
- Cost & lock-in reduction



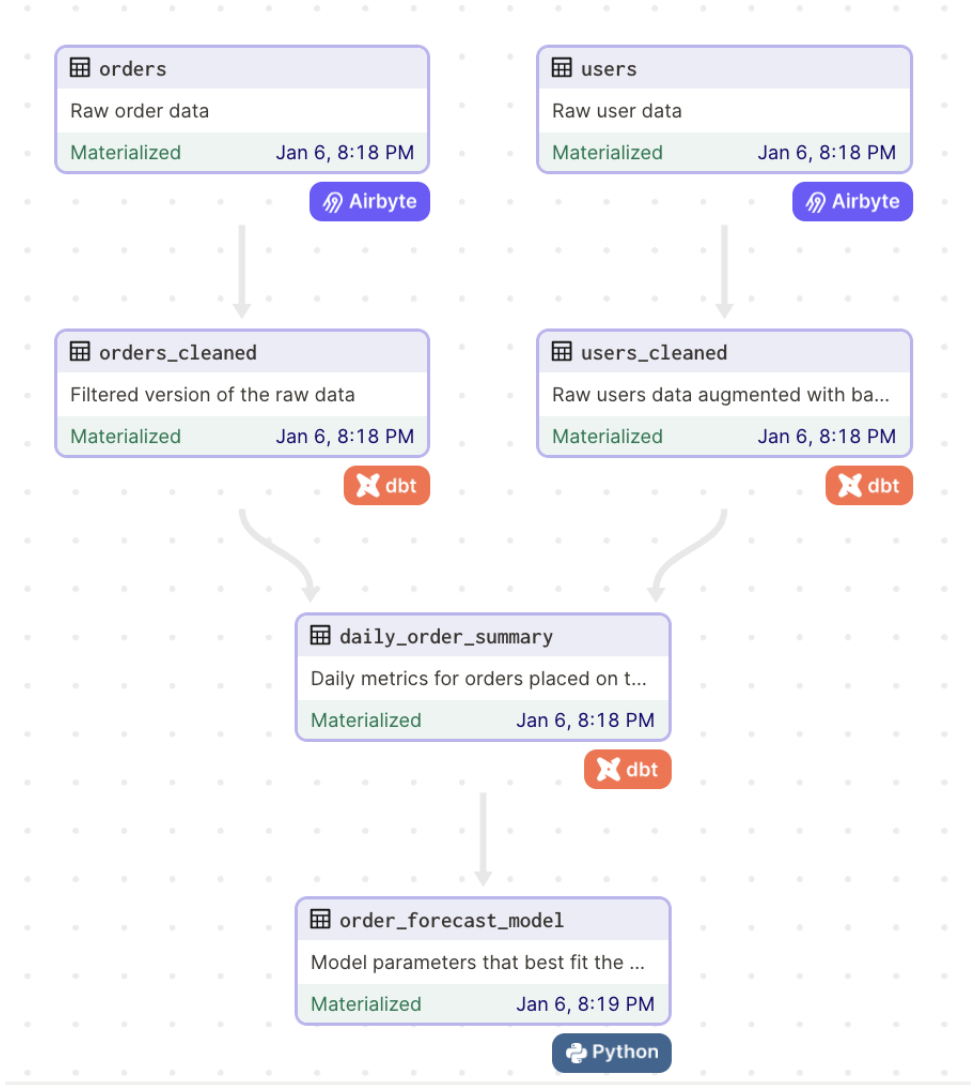
Spark at a glance



Dagster introduction

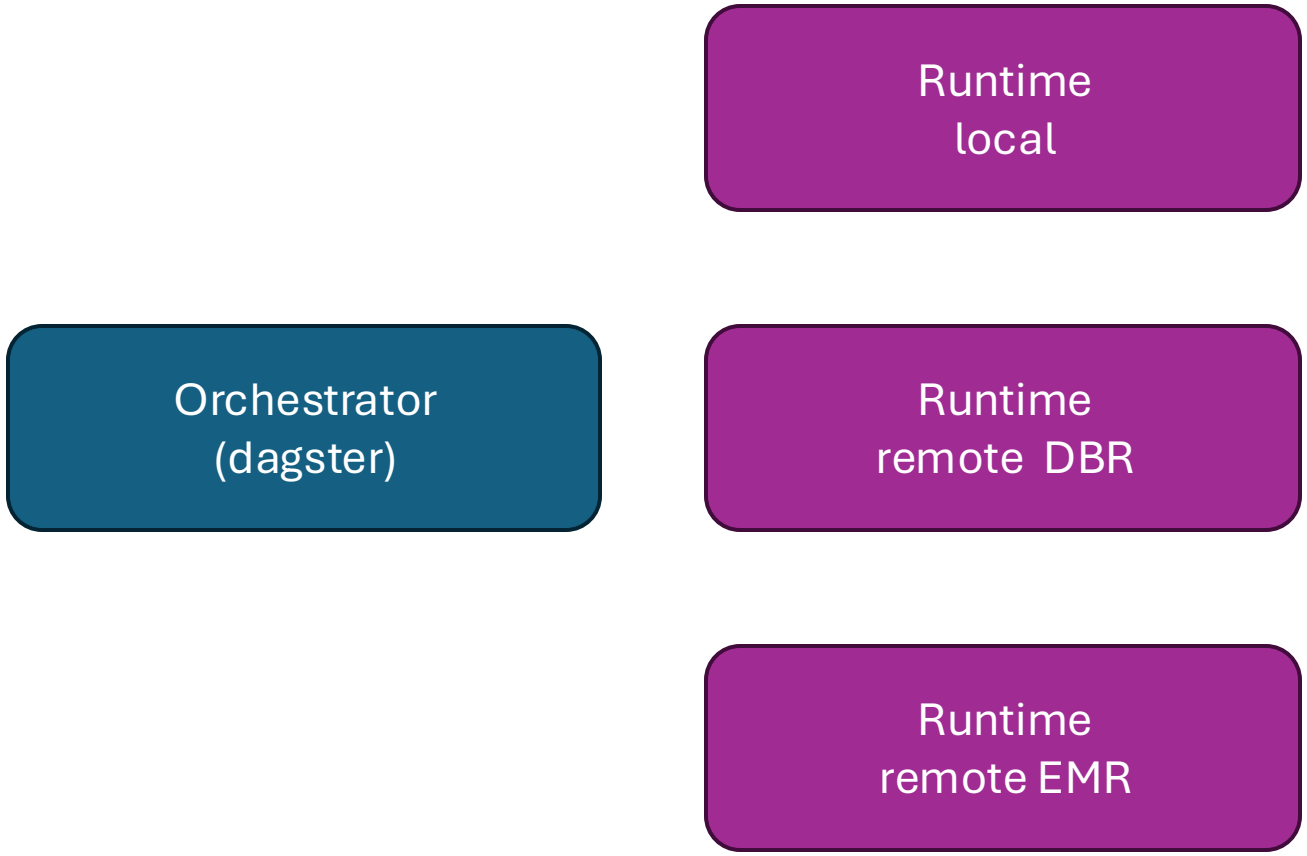
dagster

No distributed monolith of CRON strings
→ Asset aware event based orchestration

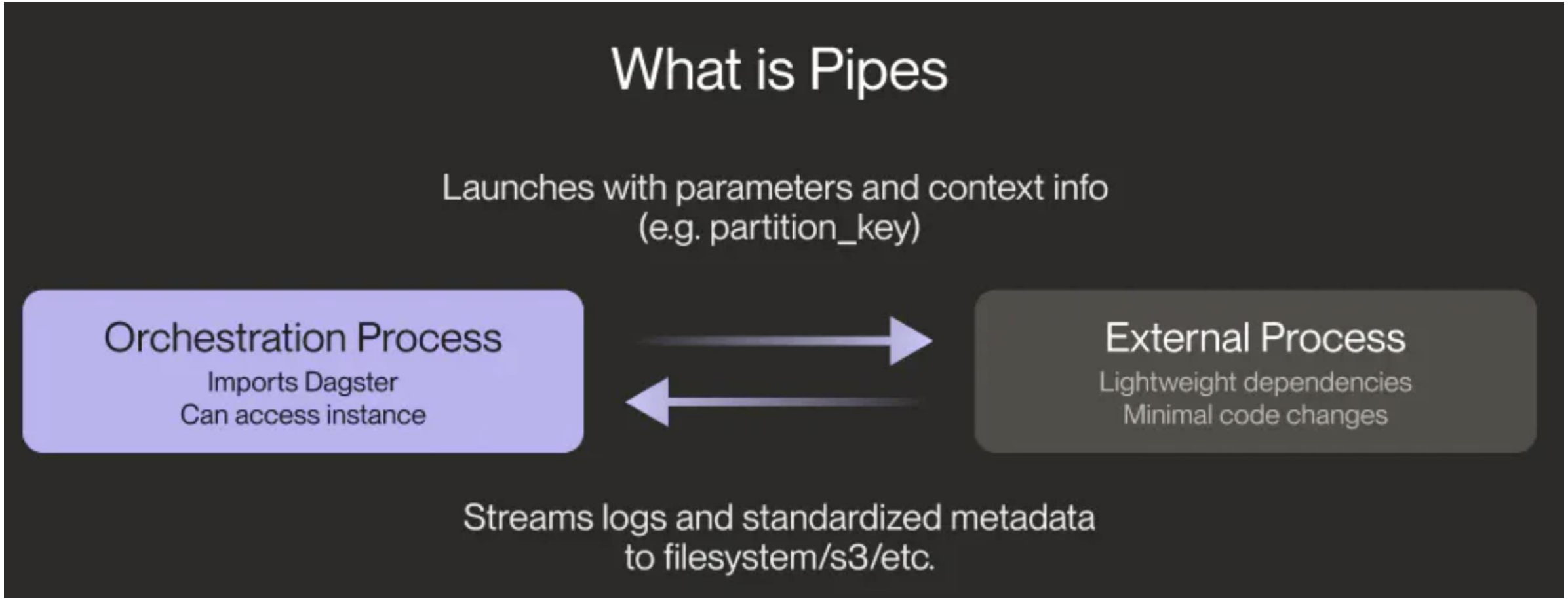


Observed challenges

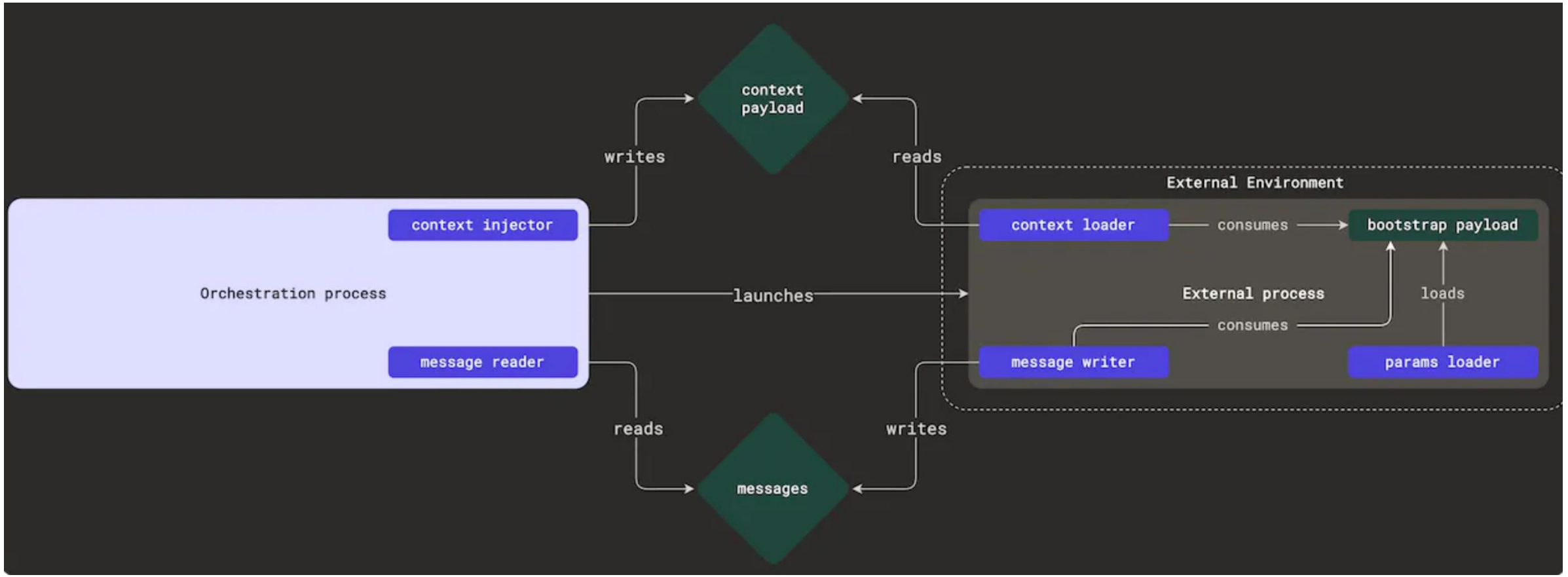
- Remote execution
  - Parameter injection
  - Logging
  - Opaque SaaS tools
  - Single pane of glass
  - Dependency bootstrap
  - Missing testability in notebooks
- Large-scale compute & orchestrator native development



Dagster-pipes



Dagster-pipes - Architecture



Dagster-pipes - Sample

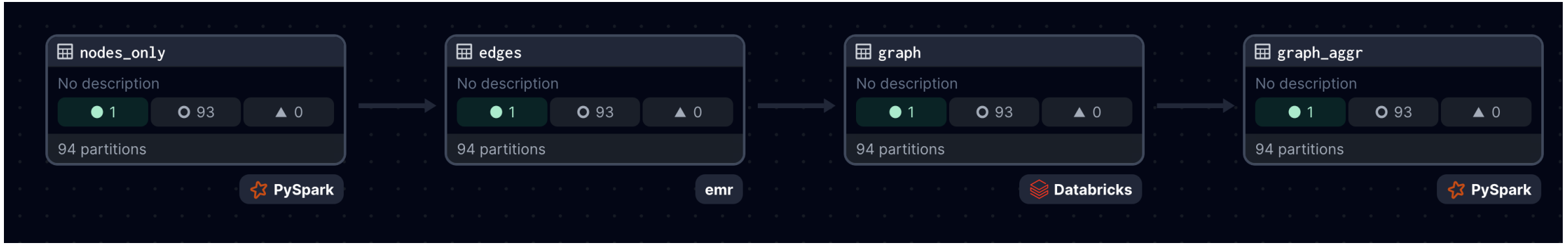
External code (with metadata)

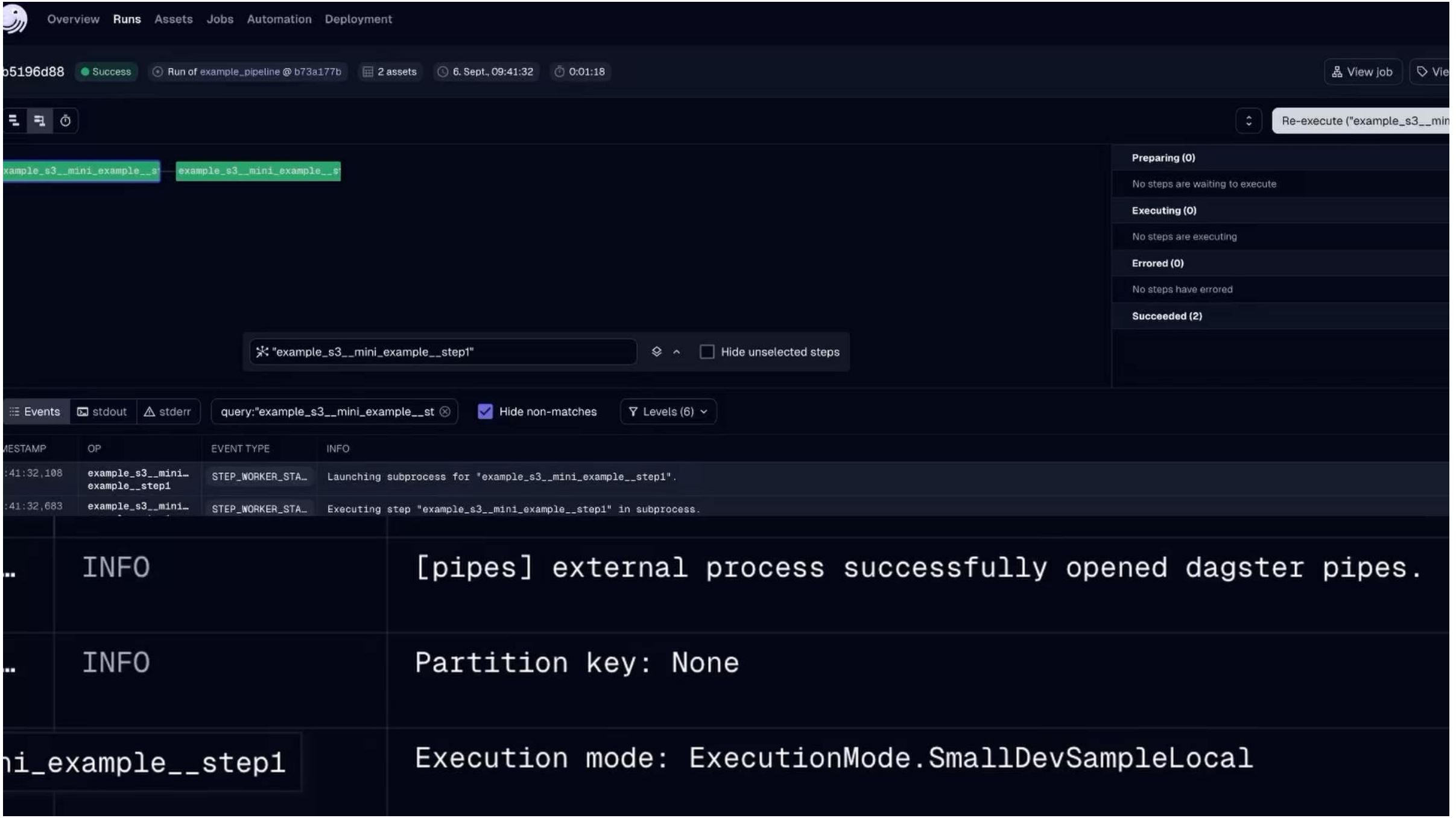
```
def main():
    orders_df = pd.DataFrame({"order_id": [1, 2
    total_orders = len(orders_df)
    context = PipesContext.get()
    print(context.get_extra("foo"))
    context.log.info("Here from remote")
    context.report_asset_materialization(
        metadata={"num_orders": len(orders_df)}
    )

if __name__ == "__main__":
    with open_dagster_pipes():
        main()
```

Internal asset shim orchestrating the execution of external script

```
@asset
def subprocess_asset(
    context: AssetExecutionContext, pipes_subprocess_client: PipesSubprocessClien
) -> MaterializeResult:
    cmd = [shutil.which("python"), file_relative_path(__file__, "external_code.py
    return pipes_subprocess_client.run(
        command=cmd,
        context=context,
        extras={"foo": "bar"},
        env={
            "MY_ENV_VAR_IN_SUBPROCESS": "my_value",
        },
    ).get_materialize_result()
```

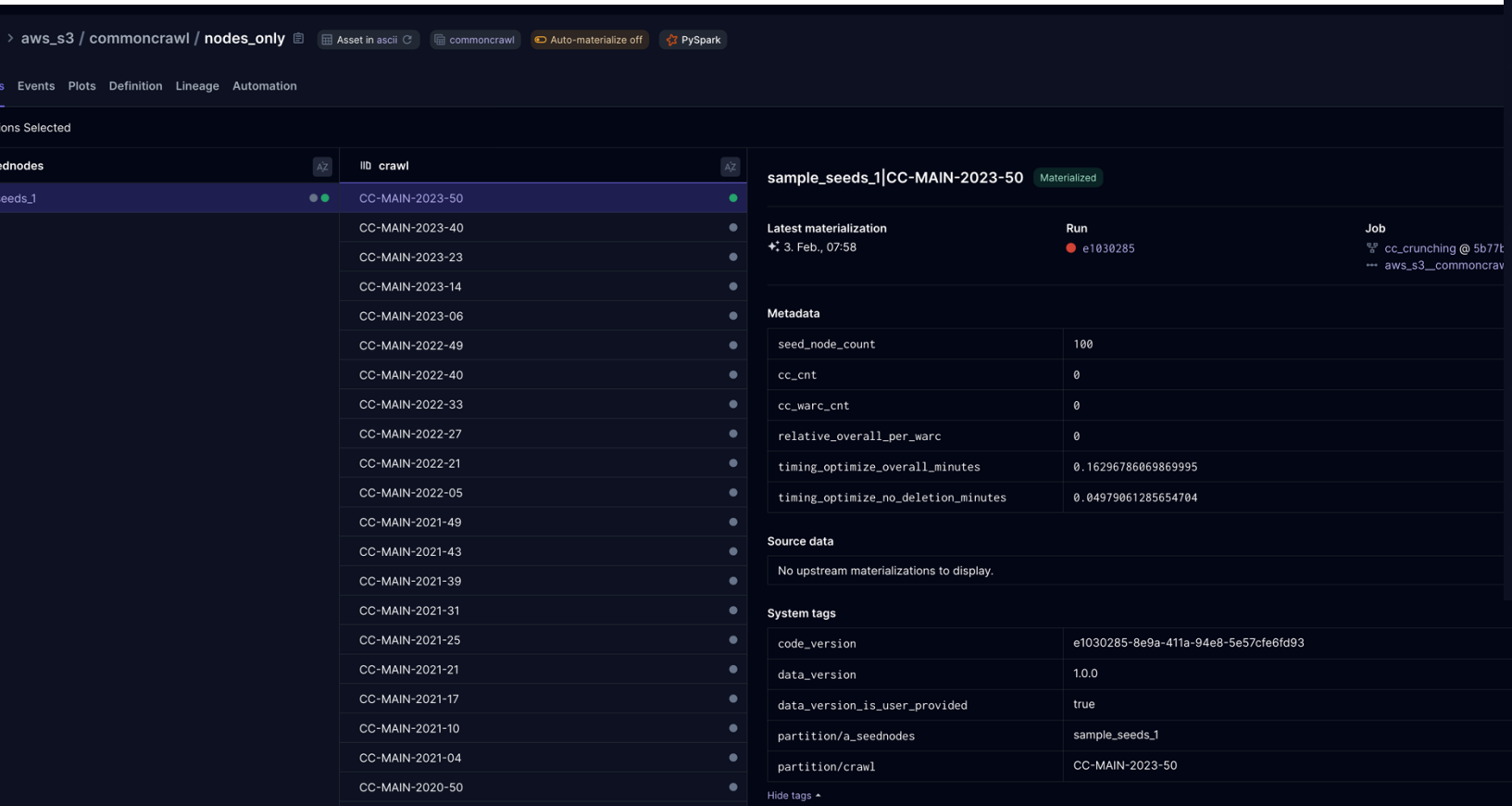
Results & Demo

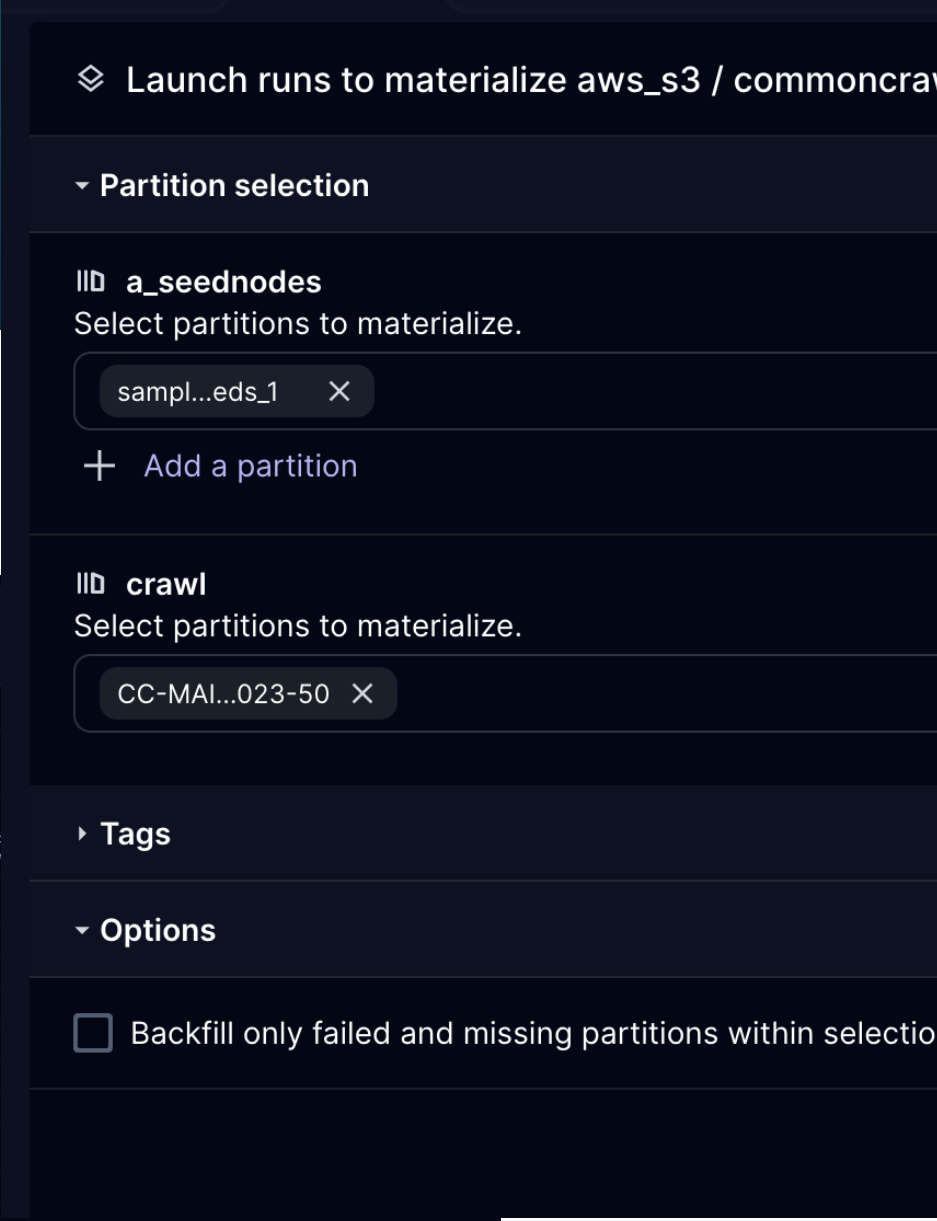




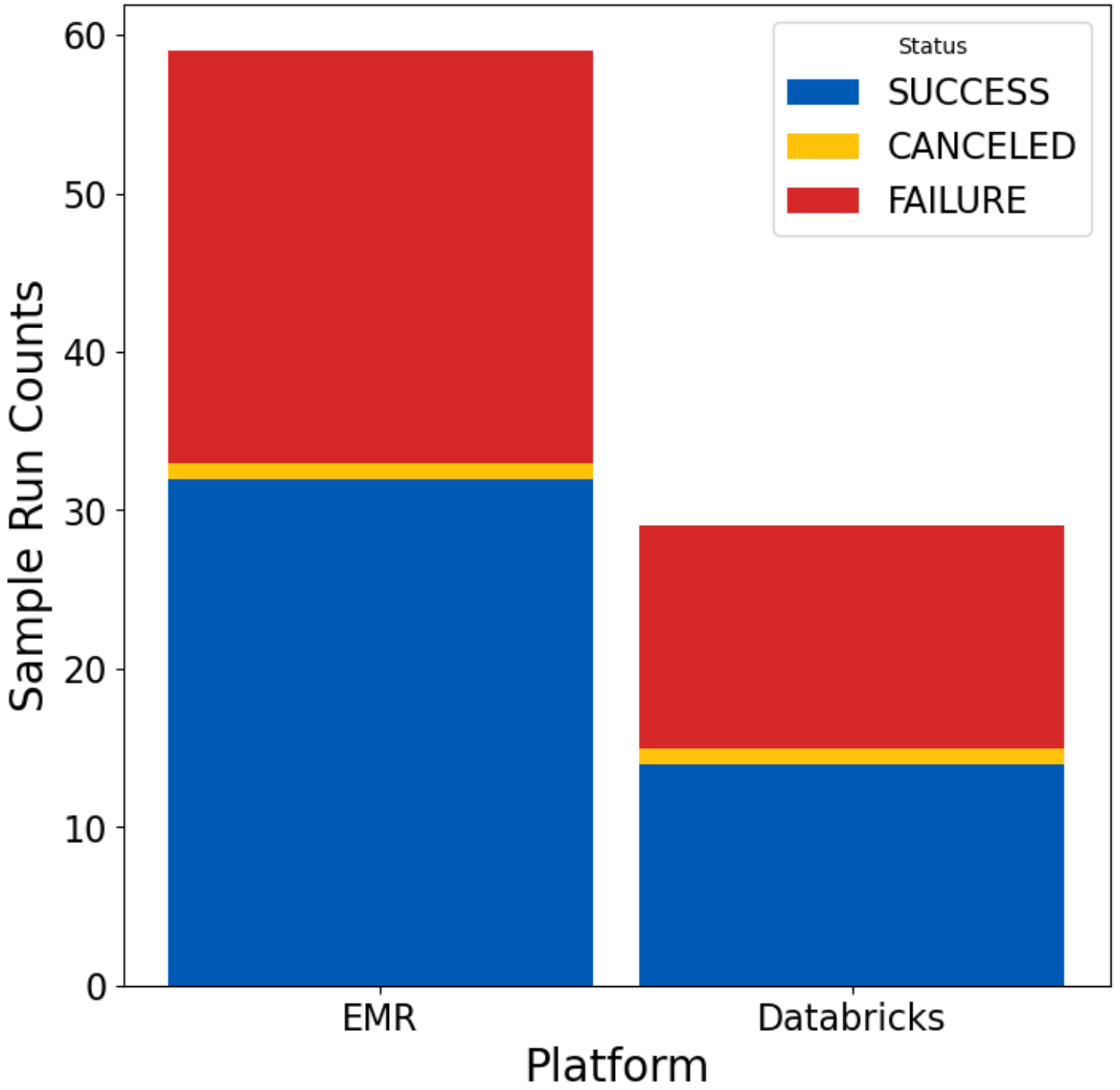
Demo: youtube.com/watch?v=W27C5LpdEkE

Partitioned UI

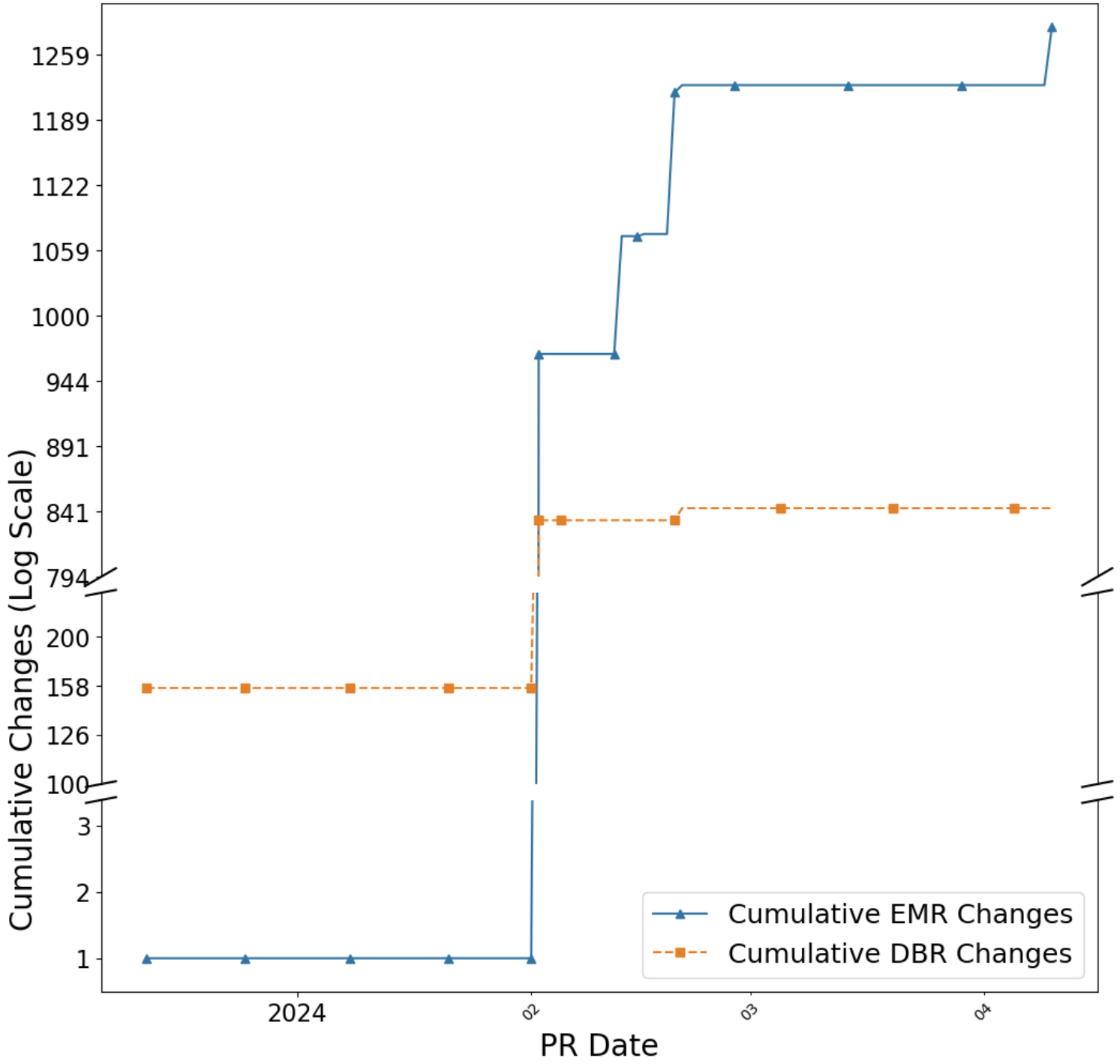




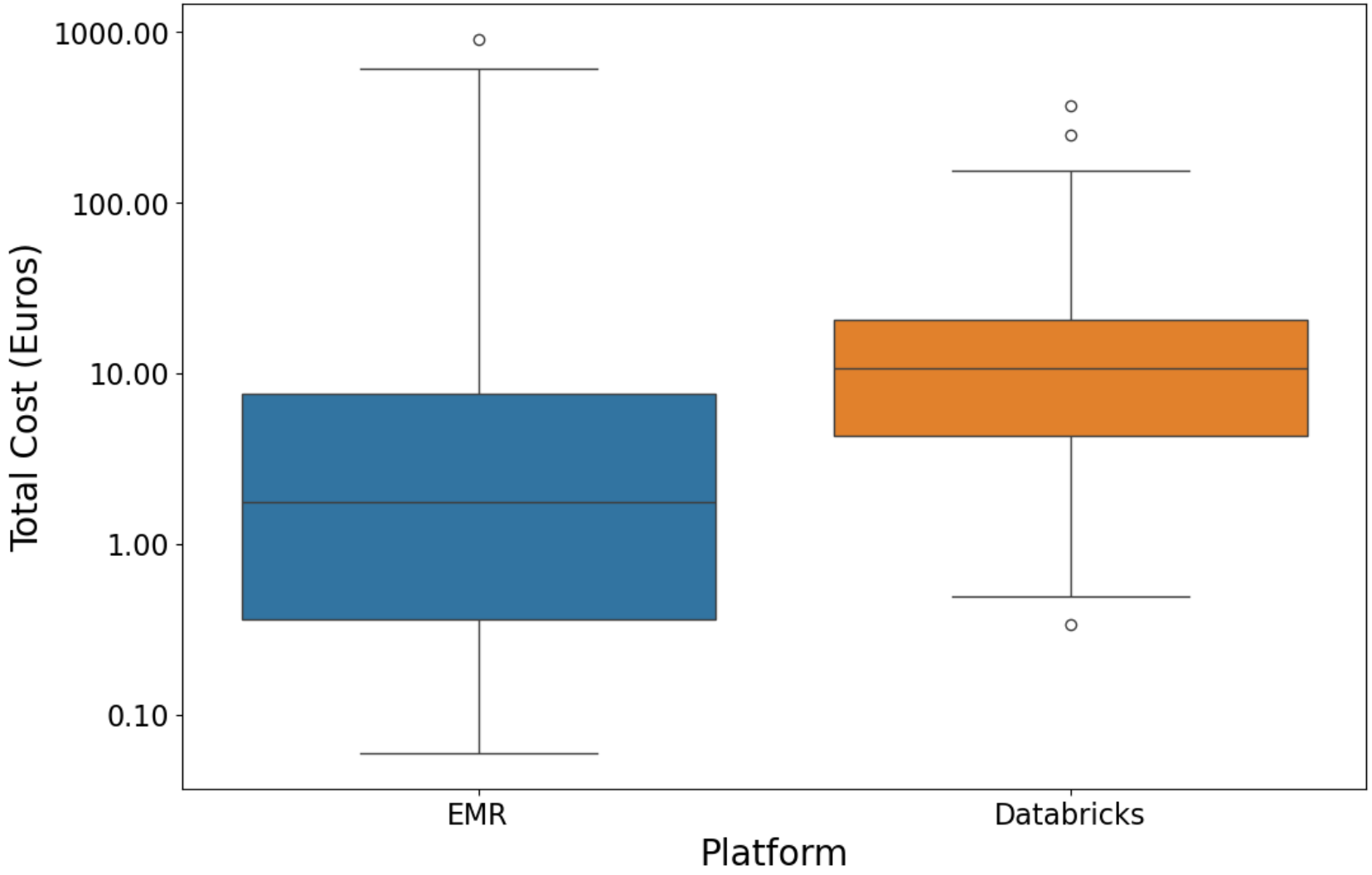
Implementation time of DBR is lower



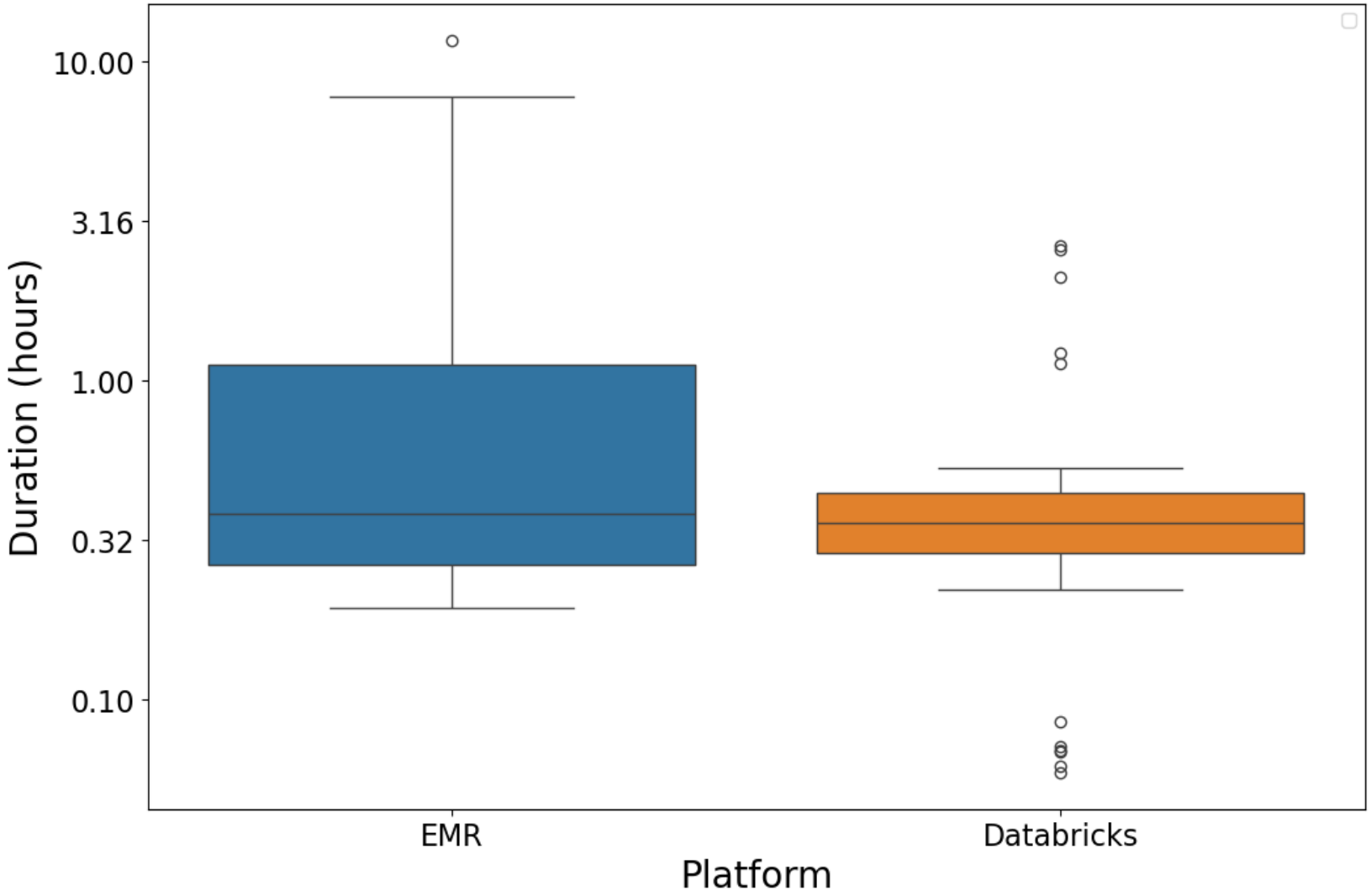
Implementation complexity of DBR is lower more & more frequent commits for EMR integration



Median cost of DBR is higher than EMR



Variability of execution time of DBR is lower



Implementation lessons

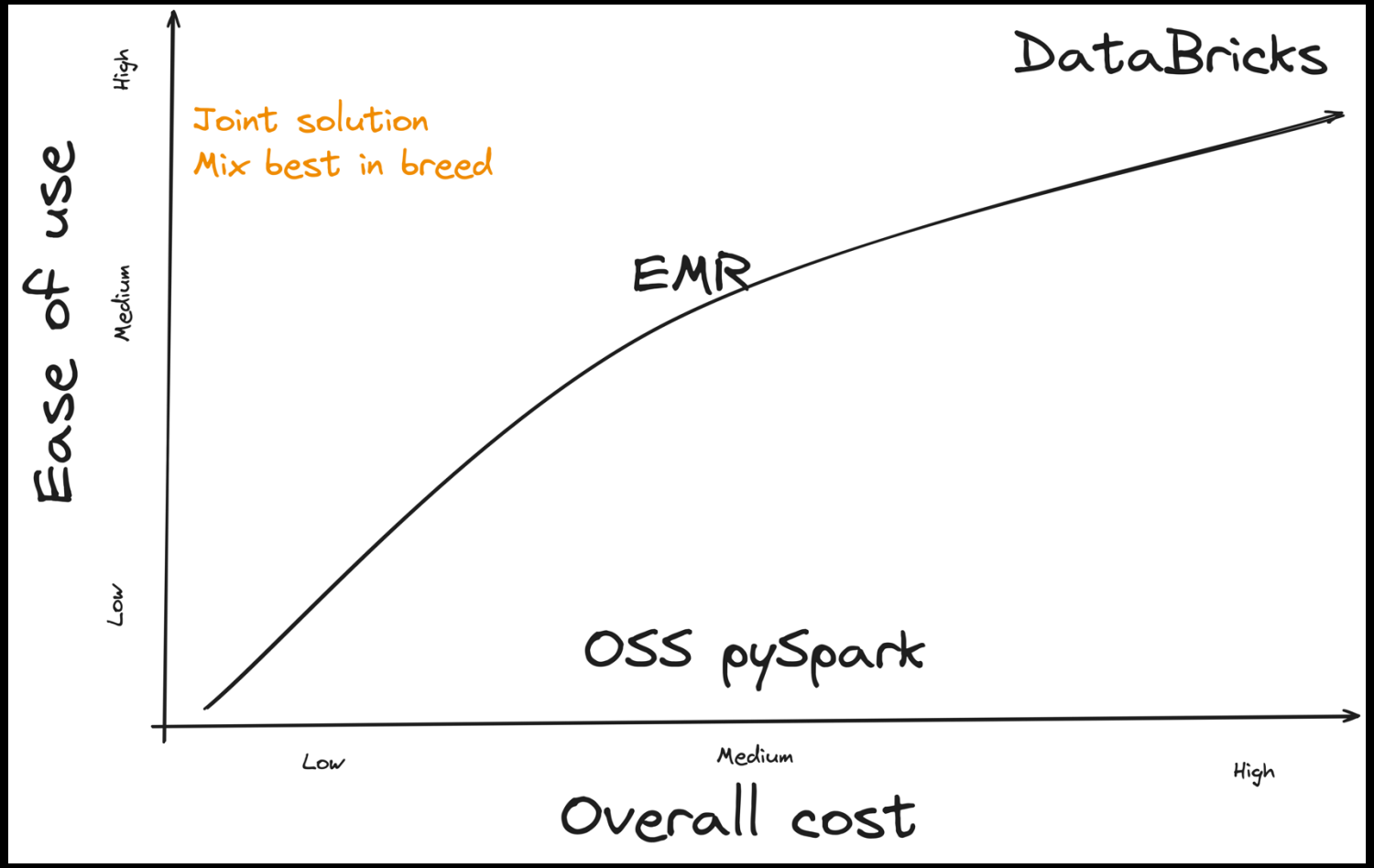
- Complexity of AWS EMR: Many low level details about AWS, spot instances, networking required (master on spot instance => 💥💥)
- Abstracting the PaaS requires deep understanding of their APIs

Tips

- maximizeResourceAllocation
- LZO
- Delta zorder on partition
- spark.databricks.delta.vacuum.parallelDelete.enabled=true

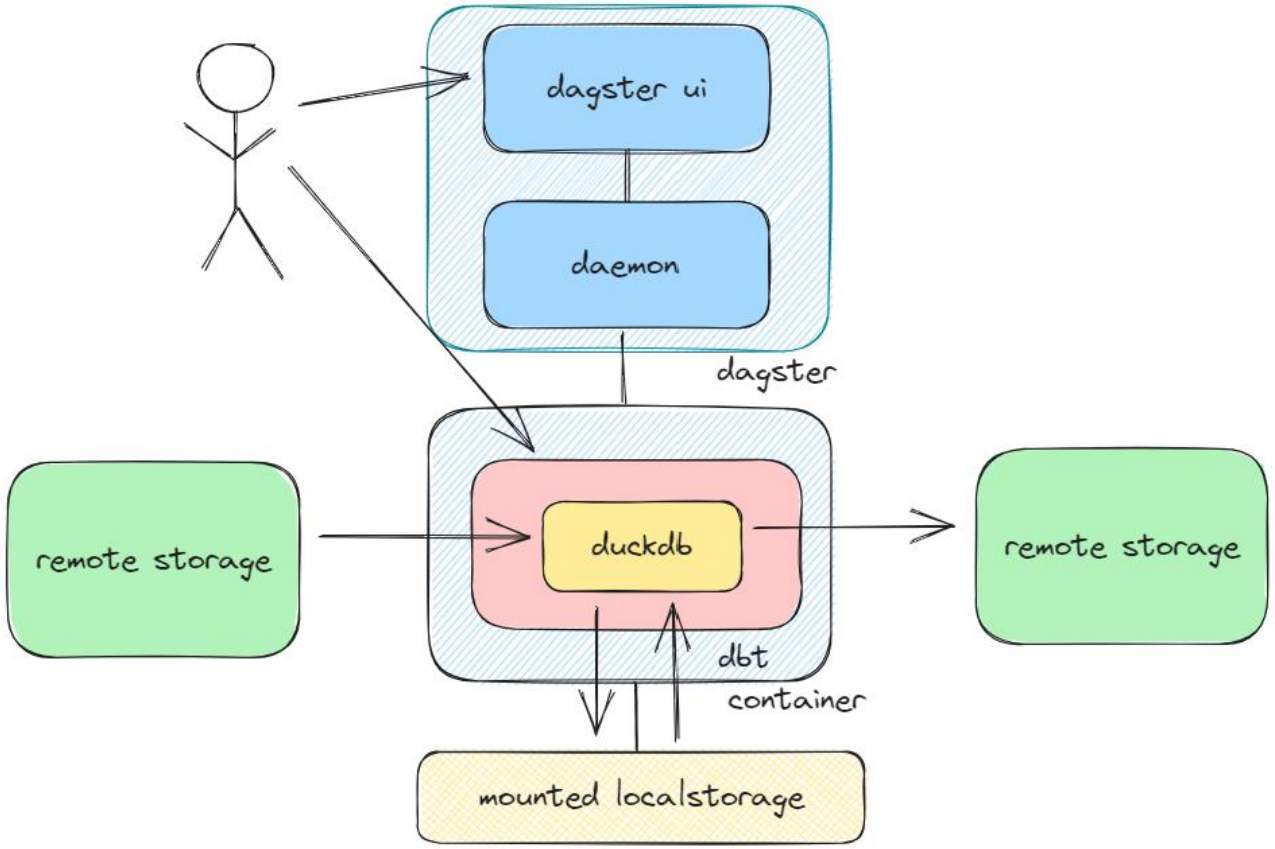
Summary

- Money saved – 43%
- Bring back software engineering best practices for data
- Flexibility
  - Data PaaS as a commodity
  - Take back control
  - Best in breed
- Single pane of glass for pipelines



Takeaway – if you have a small data problem

- Pipes allows to quickly bring in existing scripts whilst retaining observability
- High code engineering practices scales well
- Full control
- Compute technology can easily be changed (i.e. duckdb, daft, …) data-engineering.expert/2023/12/11/dagster-dbt-duckdb-as-new-local-mds



COST EFFICIENCY FOR DATA

Georg Heiler

bit.ly/efficient-spark

(data-engineering.expert/2024/06/21/cost-efficient-alternative-to-databricks-lock-in arxiv.org/abs/2408.11635 github.com/ascii-supply-networks/ascii-hydra/tree/main/src/pipelines/ascii_library_demo)