



Lessons Learned from Migrating to Airflow @ LinkedIn

Arthur Chen, Staff Software Engineer, LinkedIn
Trevor Devore, Staff Software Engineer, LinkedIn

3.0

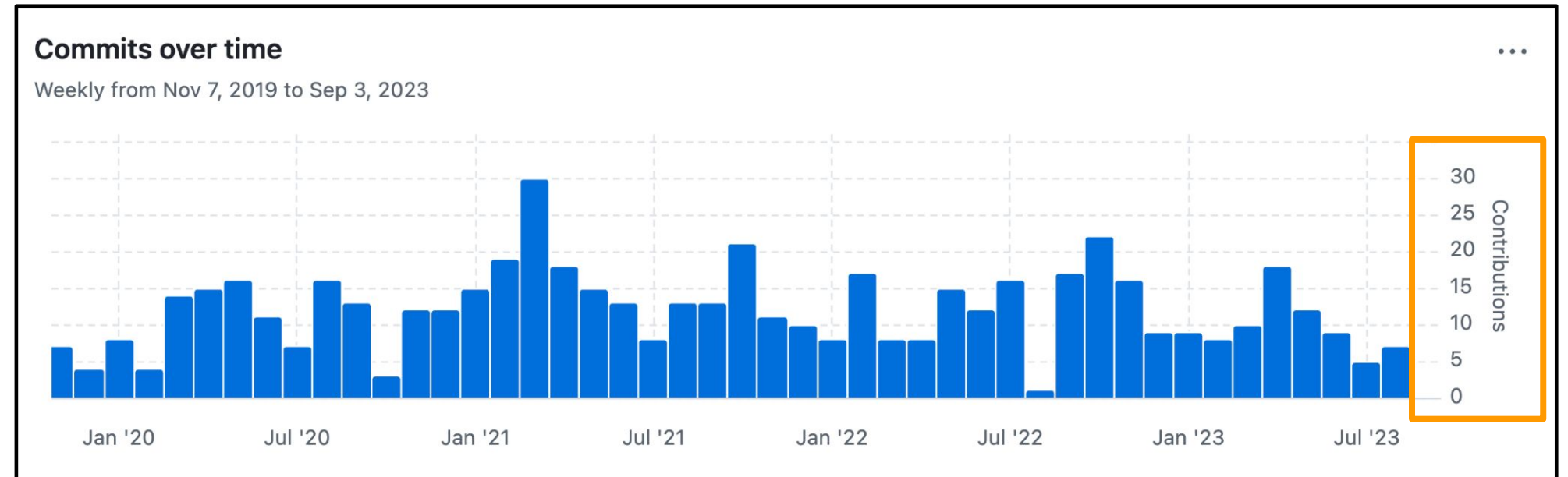


Background

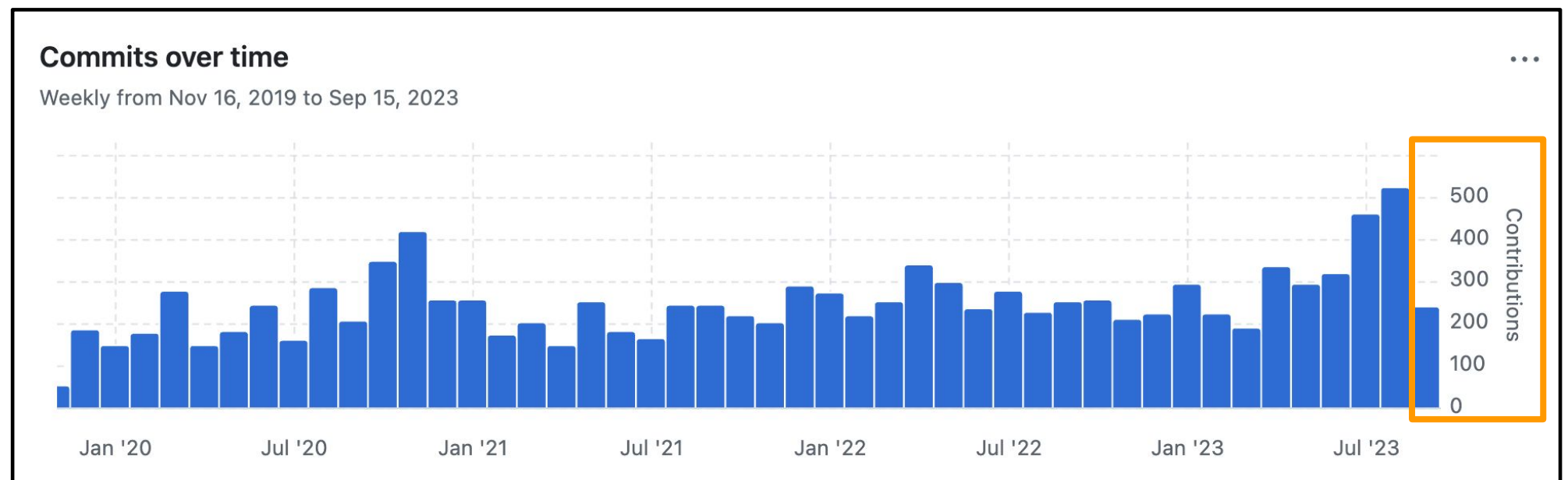
- LinkedIn runs over 100K data pipelines
 - ETL, ML, Ingestion, DataQuality, etc.
- Execute jobs across many systems
 - Spark, Trino, Pinot, etc.
- Azkaban was LinkedIn's in-house data pipeline orchestrator
 - Centralized cluster management
 - Federated by use case
 - Job Operator ecosystem (e.g. Spark, ML)
 - Groovy DSL

Why the Interest in Airflow?

- Highly Regarded in the Industry
- Rich Feature Set
 - Sensor based data trigger
 - Timezone/interval aware
 - Python DAG Authoring
- Active Community
 - Attuned to Industry Needs
 - Rapid Feature Velocity



Azkaban Github Commits



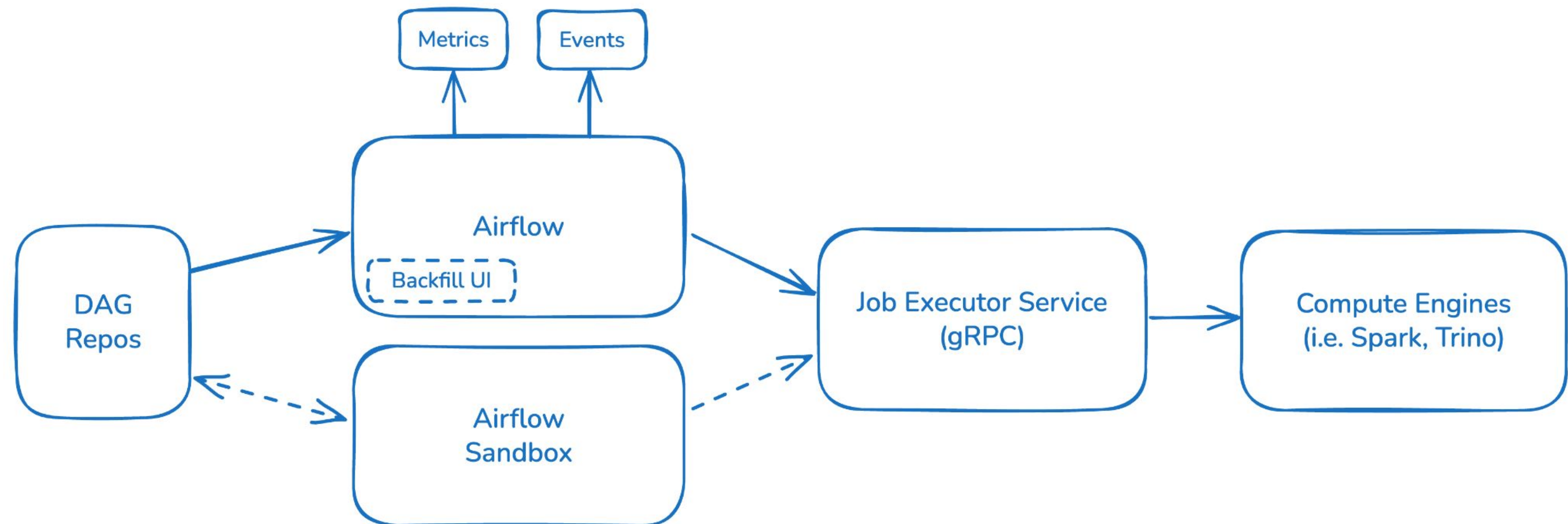
Airflow Github Commits



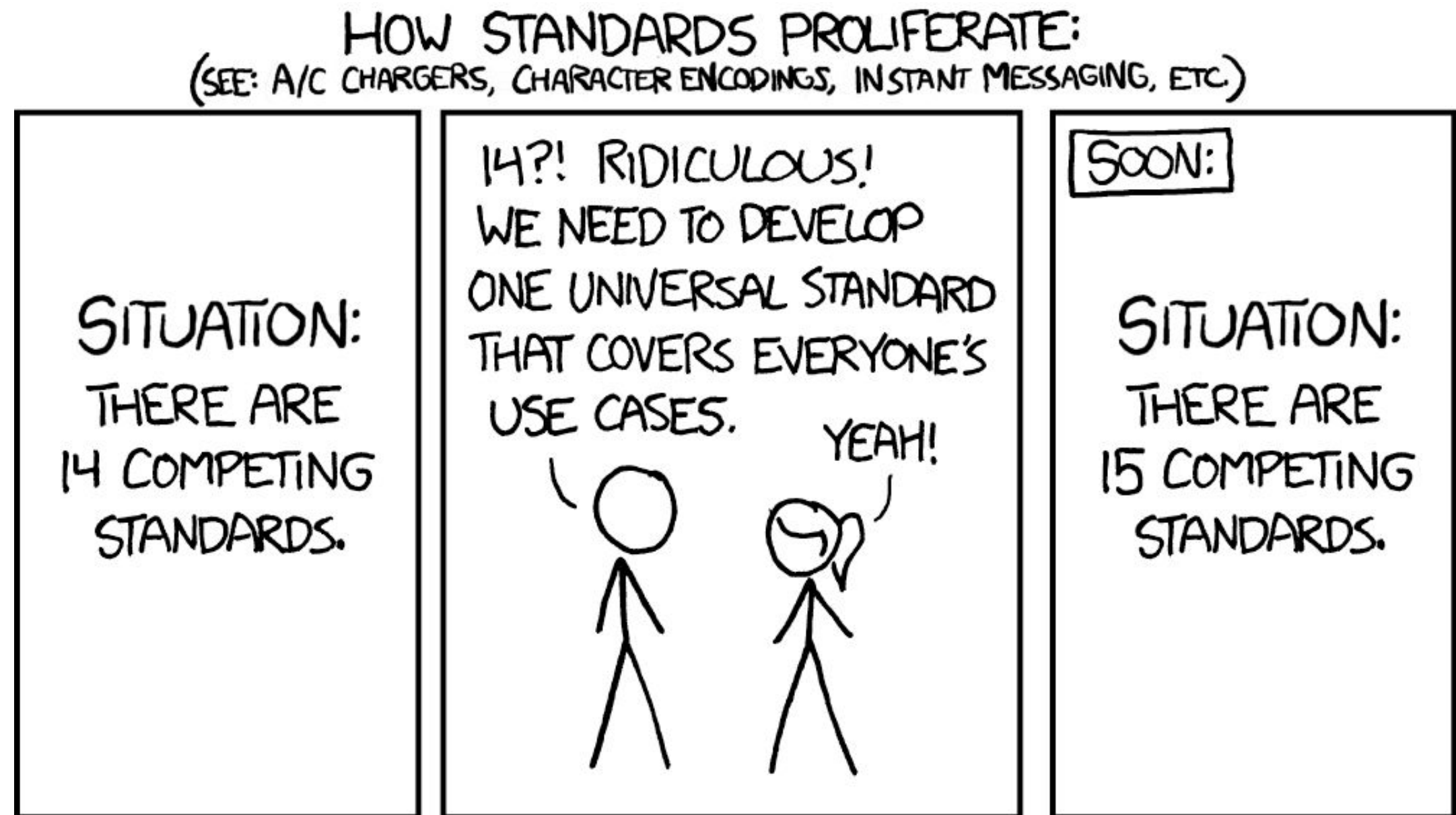
Airflow Adoption

- Evaluate Airflow for Data Pipelines at LinkedIn
- Positive feedback from early users
- Invest in Airflow as a First-Class Orchestrator
 - Practicality Considerations
 - Centralized Cluster Management
 - Preserve Job Operator Ecosystem (Java)
 - Optimized User Experience
 - Single Pane of Glass
 - Iterative Testing Experience

Architecture



- Twice the Orchestrators, Twice the Support



<https://xkcd.com/927/>

Lessons Learned

1. Build

Multi-Tenancy
in Airflow is
not Free

2. Migrate

Migrating
User Code is
Hard

3. Operate

How to Run
Airflow @
Scale



Lesson 1: Multi-Tenancy in Airflow is not Free

What does it take to provide a centralized Airflow platform for all LinkedIn use cases?

- 1000+ DAG Repos
 - DAG Directory Management
 - Distributed Validations
 - Customized one-click Deployment
 - Task Runtime Dependency Management
- Testing needs to fast, intuitive & safe
 - Ephemeral Sandbox Test Environment
 - Out-of-the-box Test Configuration
- Provide Centralized Utilities
 - Promote Airflow Best Practices
 - Enable backfills & alerting



Lesson 2: Migrating User Code is Hard

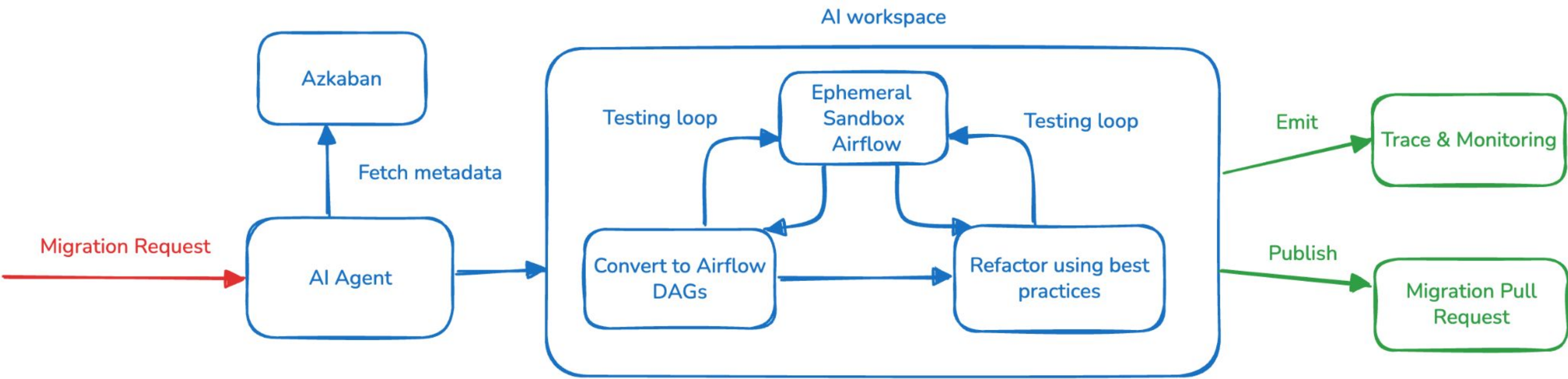
How do we migrate 100K production DAGs with:

- minimal developer effort
- zero site impact

Leverage GenAI

- Automated migration for 100K DAGs
 - Convert Source Code
 - Add orchestration & reliability features, e.g. data sensor & callbacks
 - Test & Refactor
 - Publish PR
- Evaluation & Testing
 - Reuse sandbox environments for DAG testing
 - Isolate agent from production for safety
- AI Workflow Observability
 - Rich context for debugging via OSS frameworks

GenAI Migration Workflow





Lesson 3: How to run Airflow @ Scale

How can we make Airflow meet
LinkedIn's scale requirements?

- Decouple Compute from Orchestration
 - All compute intensive tasks are handed to downstream execution service
- Scale with Resiliency
 - Custom load testing tooling (AIP-59)
 - Observability covering cluster-wide and per-DAG issues
 - Recover from Transient System Errors
- UI optimizations
 - Don't add too many tags
 - Cache authentications

Thank You.

Arthur Chen, [linkedin.com/in/yitao-arthur-chen](https://www.linkedin.com/in/yitao-arthur-chen)
Trevor Devore, [linkedin.com/in/tdevore7](https://www.linkedin.com/in/tdevore7)

