Streamlining DAG Creation With YAML In Large Organizations

Howie Wang, Software Engineer, Apple

THIS IS NOT A CONTRIBUTION

Agenda

User Story

Challenges

Strategies

Lesson Learned

Agenda

User Story

Challenges

Strategies

Lesson Learned

User Story

Scalability

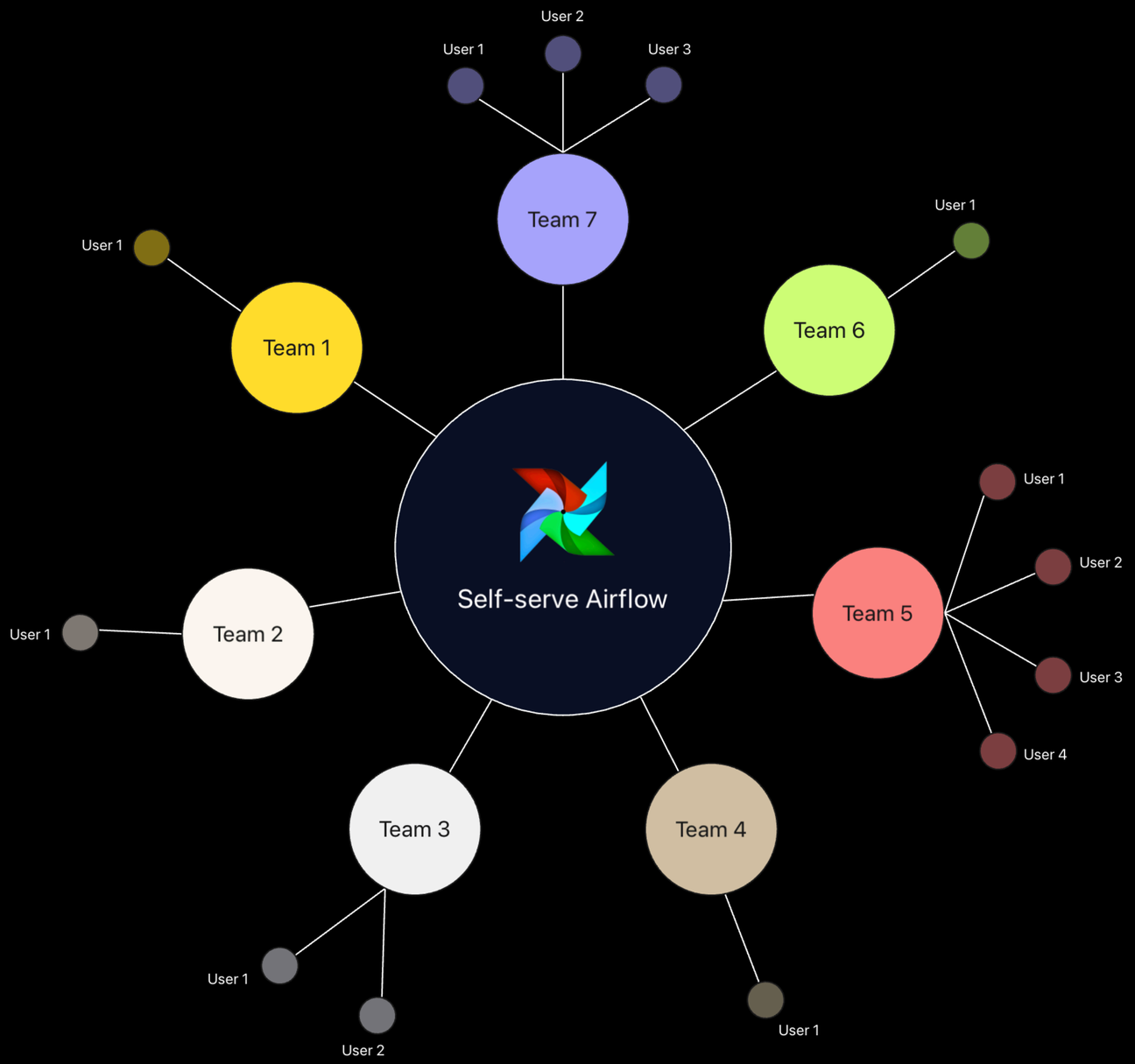
Multi-Tenancy

RBAC and Auditing

Deployment and Versioning

Monitoring and Debugging

Resource Pools



Agenda

User Story

Challenges

Strategies

Lesson Learned

Challenges

Python Expertise

Learning Airflow

Duplicative Code

Managing DAG Changes

Agenda

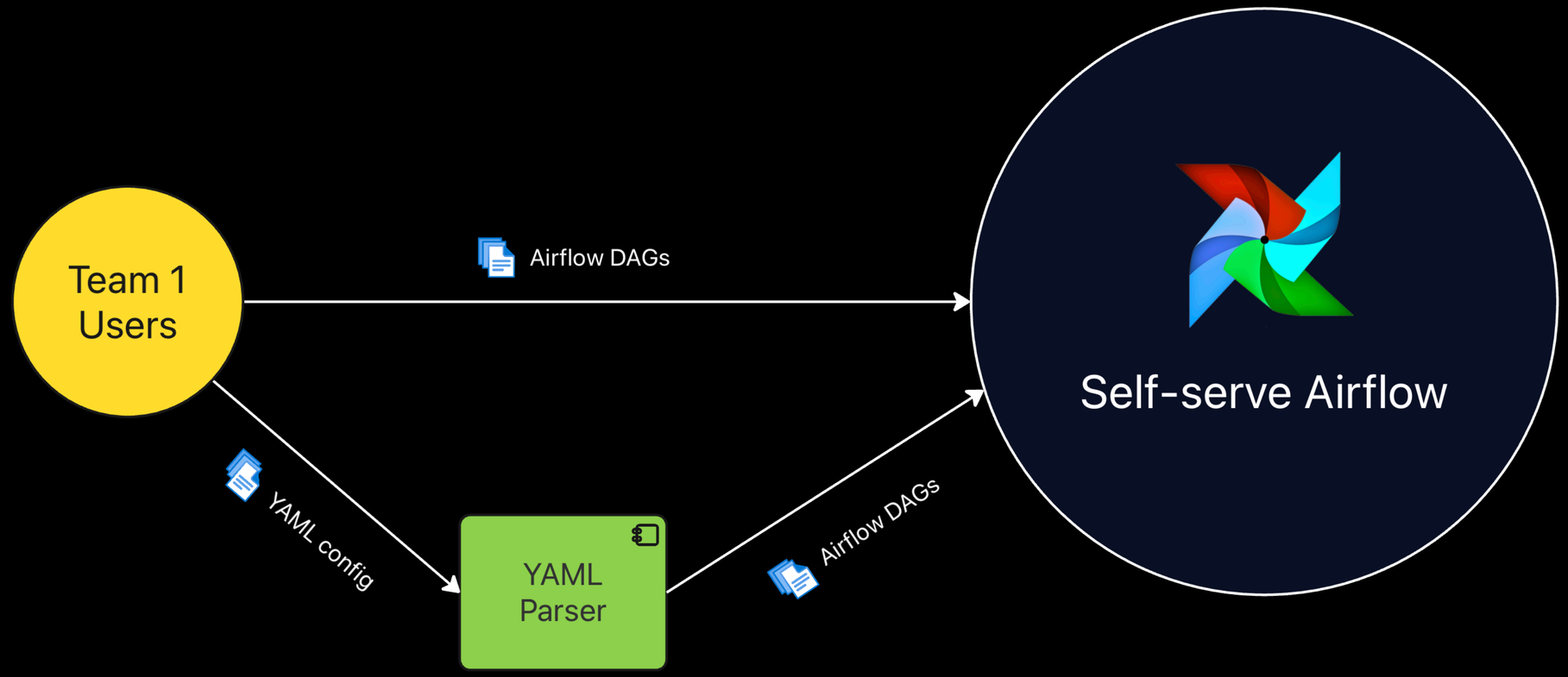
User Story

Challenges

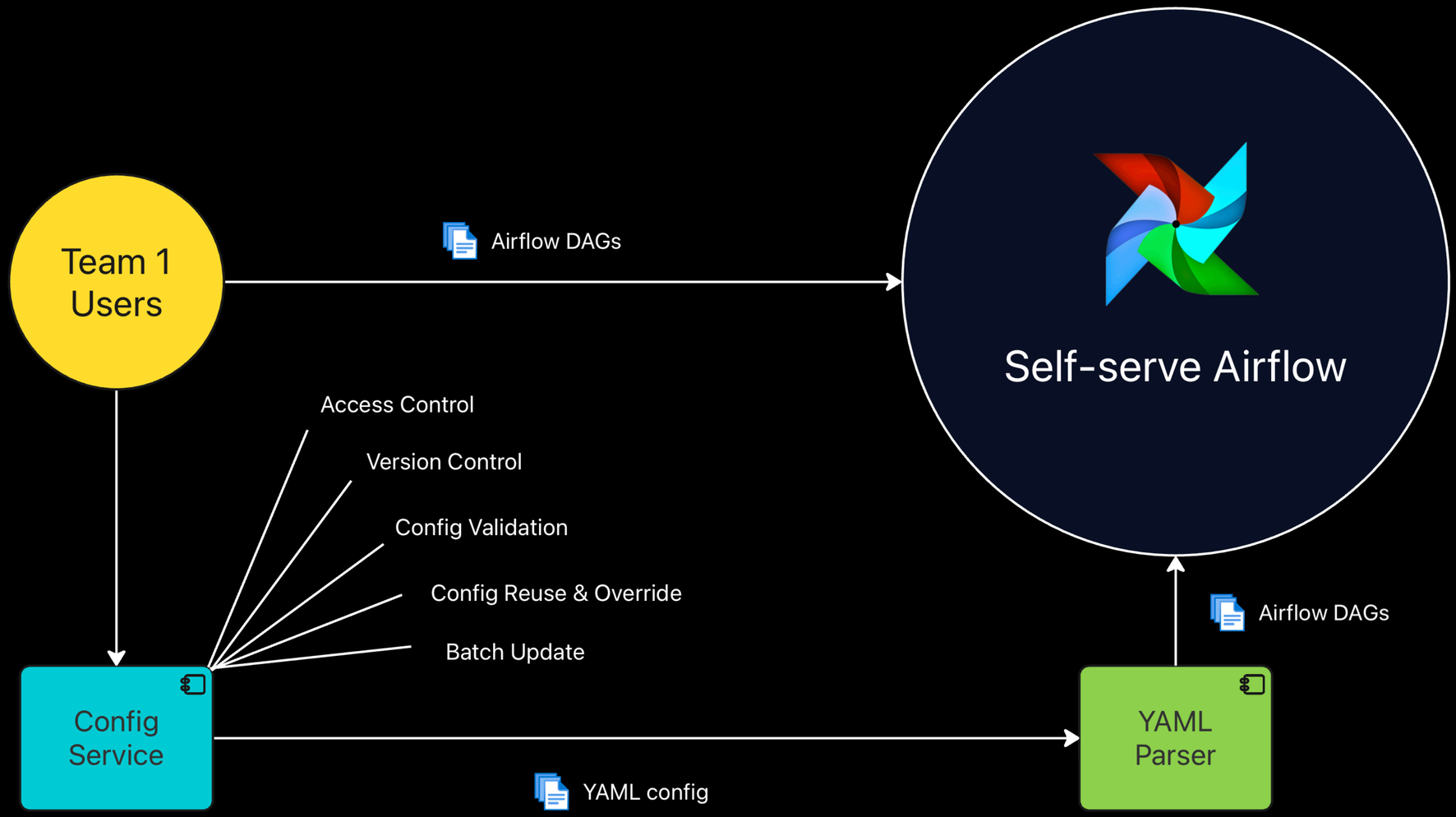
Strategies

Lesson Learned

Strategy 1: Declarative DAG via YAML



Strategy 2: YAML Config as a Service



Benefits of the YAML Approach

Simplified Syntax

more human-readable compared to Python

Standardization

enforces a consistent structure for defining DAGs

Configuration Over Code

allows for easier changes and versioning of DAGs

Better Collaboration

easier to read, review, and edit by a broader range of stakeholders

Template Reusability

use template to eliminates duplicate config for common pattern

Agenda

User Story

Challenges

Strategies

Lesson Learned

Lesson Learned

YAML DSL also has learning curve

Mapping between Airflow Operator and YAML config must be automated

YAML validator is important

Config Service can also benefit non-Airflow service

Challenges with machine generated DAGs

Q & A