УДК 004.6 **doi: 10.26906/SUNZ.2024.2.095**

В. Д. Залеський, П. С. Івановський, В. М. Федорченко

Харківський національний технічний університет радіоелектроніки, Харків, Україна

СУЧАСНІ ІНСТРУМЕНТИ ОРКЕСТРАЦІЇ ДАНИХ ДЛЯ ПОБУДОВИ КОНВЕЄРІВ АВТОМАТИЧНОЇ ОБРОБКИ ДАНИХ

Анотація. Всесвіт даних у сучасних компаніях постійно розширюється. Зі збільшенням кількості даних збільшується потреба в управлінні, синхронізації розкладів та вирішення проблем обробки. Компаніям потрібно зламати бар'єри між джерелами даних та сховищами, щоб по-справжньому використовувати всю інформацію, яку вони збирають. Оркестрація даних дозволяє організаціям автоматизувати та оптимізувати свої дані, перетворюючи їх на оперативні активи, щоб цінну інформацію можна було використовувати для прийняття бізнес-рішень у режимі реального часу. За деякими оцінками, 80% роботи, пов'язаної з аналізом даних, зводиться до збирання та підготовки даних, що означає, що оркестрація даних може скоротити велику кількість часу на обробку та планування. **Метою даної роботи** є аналіз сучасних інструментів оркестрації. **Об'єктом дослідження** є дата інженерія. **Предметом дослідження** є оркестрація даних.

Ключові слова: оркестрація даних, конвеєри обробки даних, ETL, DAG.

Вступ

Відправною точкою для аналізу даних з нуля - є сценарій, коли організація хоче максимально використовувати свої дані, але не має всіх даних про клієнтів в одному місці. Організація не може централізувати та оновлювати дані про клієнтів. Дані можуть бути ненадійними або неточними.

Оркестрація даних ідеально підходить для організацій даних з багатьма системами, оскільки не потребує масового перенесення даних до сховища. Натомість, вона надає доступ до потрібних даних, у потрібному форматі та в потрібний час. Інформацію, яка зберігається в різних сховищах, можна легко отримати та обробляти синхронно, ніби дані знаходяться в централізованому репозиторії [1].

Центральну роль в оркестрація даних відіграє ETL процес [2], він розшифровується як "вилучення (extract), перетворення (transform), завантаження (load)".

Етап вилучення передбачає підготовку перевірку цілісності та правильності даних, додавання міток та позначень або збагачення нових сторонніх даних наявними наборами даних.

Етап перетворення передбачає виявленні та виправленні (або видаленні) пошкоджених, неточних, дублюючих або аномальних даних та перетворення їх у стандартний формат.

Після того, як необроблені дані видобуто та адаптовано, вони завантажуються в цільову систему, що забезпечує синхронізацію - безперервний процес оновлення даних між джерелами та призначеннями для забезпечення їх узгодженості [3–6].

Коротка історія розвитку оркестрації робочих процесів

Оркестрація даних часто плутають з оркестрацією робочих процесів. Оркестрація робочих процесів – це процес запуску та моніторингу стану завдань. Її сутність – реалізувати аналог event-driven систем на базі пакетної обробки.

Оркестрація даних є підмножиною оркестрації робочих процесів і забезпечує надійну та ефективного синхронізацію даних у продакшен середовищі. На ряду з елементами ETL вона може також включати: елементи управління середовищем (CI/CD для даних або GitOps для даних або безперервна інтеграція та доставка даних), контроль доступу на основі ролей ("RBAC"), оповіщення та моніторинг стану даних.

Інструменти оркестрації робочих процесів існували вже протягом тривалого часу, але лише нещодавно їх почали називати інструментами оркестрації даних.

Оркестрація даних у сфері обробки даних бере свій початок від великих технологічних компаній. Luigi був створений у Spotify на початку 2010-х років. Airflow, переможець цієї битви та лідер галузі, з'явився у Airbnb. У Meta є дуже схожий внутрішній, непублічний інструмент під назвою DataSwarm.

З того часу з'явилося багато інших пакетів оркестрації робочих процесів із відкритим вихідним кодом.

На відміну від Airflow і Luigi, які дійсно були відкритими (Airflow зрештою є частиною Apache Software Foundation), Prefect, Dagster, Mage, Kestra претендують на те, щоб бути кращими, простішими та зручнішими альтернативами Airflow [7], особливо для організації даних.

По суті, всі вони дуже схожі, пропонуючи фреймворк для написання коду та виконання орієнтованих ациклічних графів (DAG) у контейнеризованому середовищі. Ці інструменти є безкоштовні, але за розгортання та додаткові хмарні функції, такі як логування, контроль доступу на основі ролей, підтримку клієнтів доведеться заплатити.

Огляд сучасних інструментів

До основних характеристик сучасних інструментів оркестрації робочих процесів з відкритим вихідним кодом можно віднести:

1) Визначення DAG за допомогою Git-контролю: інструменти дозволяють користувачам створювати та керувати DAG як кодом, використовуючи переваги Git для версіонування, історії змін та спільної роботи.

2) Інструменти містять планувальник, який виконує DAGs відповідно до визначеного розкладу або інтервалів.

3) Інструменти дозволяють запускати DAG вручну або автоматично на основі певних подій (тригерів). Вони також надають функції моніторингу для відстеження виконання DAG та оповіщення про помилки або успішне завершення. Крім того, інструменти дозволяють керувати DAG, наприклад, зупиняти, перезапускати або оновлювати їх.

4) Більшість інструментів дозволяють користувачам писати логіку DAG на Python, що забезпечує гнучкість та інтеграцію з різноманітними бібліотеками Python для обробки даних та інших завдань.

5) Користувальницький інтерфейс(КІ): хоча деякі інструменти орієнтовані на розробку за допомогою коду, багато хто пропонує КІ для візуалізації DAG, перегляду історії виконання, налаштування параметрів та керування іншими аспектами робочих процесів.

Перейдемо до безпосереднього розгляду інструментів.

Заснований у 2018 році Prefect [8] позиціонує себе як "новий стандарт автоматизації потоків даних". Зовнішній вигляд інтерфейсу користувача Prefect наведено на рис. 1.

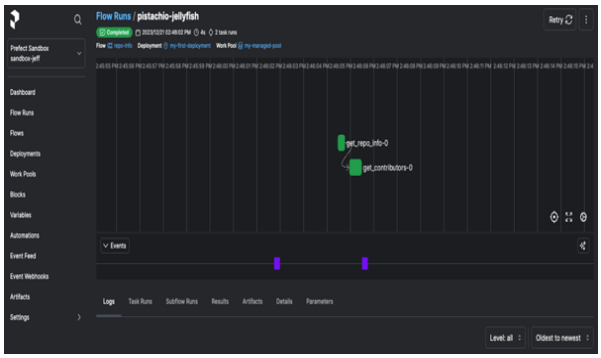

Рис. 1. Зовнішній вигляд КІ Prefect

Prefect має зручний КІ і спеціально розроблений для роботи з даними. Він пропонує простий і централізований спосіб керування та моніторингу всього стеку обробки даних. Додавши кілька декораторів до Python-коду, можно перетворити свої функції на завдання Prefect і безпроблемно інтегрувати їх у робочі процеси. Prefect доступний розробникам з базовими знаннями Python, що полегшує початок оркестрації конвеєрів обробки даних.

Prefect пропонує унікальну "гібридну модель" для оркестрації робочих процесів, яка вирішує критично важливу проблему, яка постає при використані хмарних рішень: довіра. Традиційні служби оркестрації вимагають запуску коду клієнта на інфраструктурі провайдера. Це створює проблему довіри: клієнти повинні довіряти постачальнику свій конфіденційний код, а провайдери повинні гарантувати, що код не є зловмисним. Локальні рішення стикаються з аналогічною проблемою довіри, хоча й у протилежному напрямку (клієнти довіряють програмному забезпеченню постачальника).

Prefect Cloud - сервіс, зосереджений виключно на оркестрації, а не на виконанні коду. Це усуває необхідність для клієнтів довіряти Prefect свій код. Клієнти зберігають повний контроль і безпеку: їх код залишається на їх приватній інфраструктурі.

Як працює гібридна модель Prefect?

Клієнти проектують і тестують робочі процеси за допомогою Prefect Core на власній інфраструктурі. Після завершення робочий процес реєструється в Prefect Cloud. При цьому передаються метадані про робочий процес (завдання, залежності, розклад тощо) - не сам код. Prefect Cloud використовує отримані метадані для оркестрації робочого процесу без необхідності в коді. Він керує плануванням розкладу, контролює стани виконання (запуск, успіх, помилка, повторний запуск) і координує кілька паралельних виконань. Далі Prefect Agent, що працює на інфраструктурі клієнта, виконує завдання локально або у віддаленому кластері. та передає стан виконання назад до Prefect Cloud.

Розроблений у 2021 році, Mage [9] - інструмент оркестрації, спеціально розроблений для ETL-конвеєрів даних. Зовнішній вигляд інтерфейсу користувача Mage наведено на рис. 2.

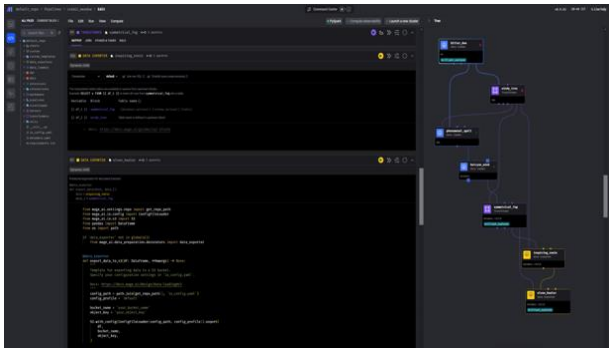

Рис. 2. Зовнішній вигляд КІ Mage

Він працює за допомогою блоків, які можуть мати декілька залежностей від попередніх або наступних блоків. Кожен блок має свій ізольований код, призначений для легкого повторного використання в різних конвеєрах.

Mage пропонує зручний КІ у вигляді інтерактивного блокнота для легкої розробки. Блокноти підтримують розробку кода на Python, R та SQL і дозволяють не турбуватися надто про обробку винятків, оскільки Mage бере це на себе. За допомогою Mage можна встановлювати зв'язки між різними блоками конвеєра безпосередньо через інтерфейс, що усуває потребу в додатковому коді.

Одна з переваг використання Mage - це можливість переглядати та візуалізувати результати вашого конвеєра даних в режимі реального часу. Це дозволяє перевіряти та аналізувати дані без очікування розгортання в шарі оркестрації.

На додаток до розробки конвеєрів для пакетної обробки, Mage також дозволяє створювати конвеєри для стрімінгу даних. Наразі Mage підтримує Kafka, Azure Event Hub, Google Cloud PubSub, Kinesis та RabbitMQ сервіси для роботи зі стрімінгом даних, що

дозволяє обробляти та аналізувати дані в режимі реального часу.

Слід зазначити що, для запуску повного набору інструментів оркестрації іншим аналогам може знадобитися кілька Docker-контейнерів. Це ускладнює роботу на окремому без серверному Docker-екземплярі, такому як Google Cloud Run або аналогічному Azure Container Instances. Mage робить акцент на простоті, він працює на одному Docker-контейнері, що значно полегшує його розгортання.

Kestra [10] розроблена у 2022 році для полегшення масштабування оркестрації робочих процесів. Вона здатна обробляти збільшення робочого навантаження, дозволяючи розподіляти завдання між декількома вузлами кластеру. Зовнішній вигляд інтерфейсу користувача Kestra наведено на рис. 3.

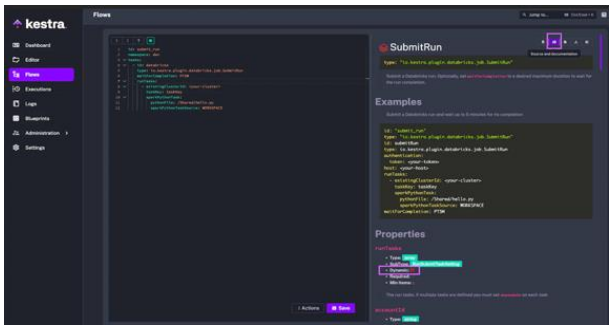

Рис. 3. Зовнішній вигляд КІ Kestra

Останнім часом багато хто порівнює Kestra з Terraform, оскільки обидва інструменти використовують DSL для керування та автоматизації робочих процесів. Terraform справив значний вплив на DevOps, встановивши стандарт IaC. Kestra прагне запровадити схожі принципи в сфері оркестрації та автоматизації, хоча й з деякими відмінностями в підході.

Цей декларативний підхід означає, що визначається бажаний кінцевий стан, а не кроки для досягнення цього стану. За допомогою визначення робочих процесів у конфігураційному файлі YAML, Kestra абстрагується від складнощів процедурного коду, що робить створення, розуміння та підтримку робочих процесів легшими.

Хоча Kestra використовує YAML для визначення робочих процесів, вона пропонує свободу використання будь-якої мови програмування для написання скриптів всередині робочих процесів. Ця незалежність від конкретної мови дозволяє розробникам використовувати свої наявні навички, чи то Python, R, або інші мови.

Дизайн Terraform зосереджений на модульних конструкціях, які дозволяють розділяти складні ресурси на повторно використовувані компоненти. Kestra підтримує цю філософію модульності. За допомогою таких функцій, як креслення (blueprints) та підпроцеси (subflows), вона пропонує структурований підхід до створення робочих процесів, що покращує їх повторне використання та обслуговування. Особливо виділяються підпроцеси Kestra, оскільки вони дозволяють повторно використовувати частини робочих процесів у різних операціях, полегшуючи оновлення та зміни в єдиному централізованому компоненті.

Заснований у 2022 році, Dagster [11] моделює конвеєри даних з точки зору активів даних, які вони виробляють та споживають. Зовнішній вигляд інтерфейсу користувача Dagster наведено на рис. 4.

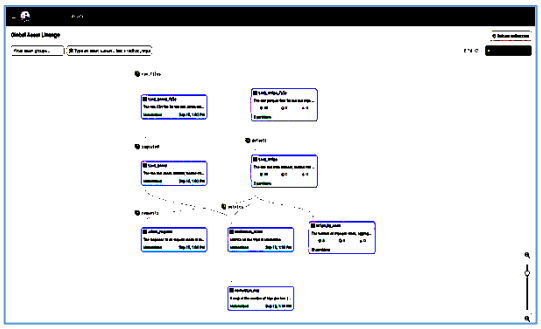

Рис. 4. Зовнішній вигляд КІ Dagster

Актив - це об'єкт у постійному сховищі, наприклад, моделі dbt, таблиці Snowflake або навіть CSV-файли. Програмно-визначений актив - це опис в коді, який повинен існувати і як створювати та оновлювати цей актив. Програмно-визначені активи дозволяють використовувати декларативний підхід до управління даними, в якому код є джерелом істини щодо того, які активи даних повинні існувати і як вони обчислюються.

Dagster вважає, що оркестратор повинен бути обізнаним щодо активів. Ця інформація використовується для побудови Каталогу активів Dagster. Це принципово нового погляду на роботу оркестратора, який пов'язує активи з обчисленнями, що їх генерують. Користувачі можуть переходити до оркестратора, шукаючи створені активи, а не конвеєри, які їх створили.

Для визначення робочого процесу у Dagster користувач пише декоровані функції Python, що визначають обчислення та структуру графа. Подібно до Airflow, реалізація окремого вузла графа може робити все, що може Python. Проте, Dagster вважає, що функції повинні офіційно декларувати свої вхідні та вихідні дані, надавати гарантії типізації для цих даних, визначати необхідну конфігурацію тощо.

Dagster API дозволяє розділити обов'язки між обчисленнями та вводом/виводом. Це необхідна умова для тестування програм для роботи з даними, що забезпечують швидкий зворотній зв'язок для розробників. Наприклад, у режимі "тестування" ресурс може зберігати фрейм даних у файлову систему, а ресурс у режимі "виробництво" може зберігати фрейм даних у сховищі об'єктів, такому як S3

Dagster є достатньо гнучким для виконання обчислень без необхідності спеціальної інфраструктури. Не потрібна інфраструктура, розклад або реєстрація стану для виконання конвеєрів. Структура графа та розклад виконання є розділеними концепціями: розклад та сенсори визначаються незалежно від конвеєра.

Порівняння інструментів оркестрації даних. Порівняльна таблиця інструментів наведена у табл. 1.

Таблиця 1 – **Порівняльна характеристика інструментів оркестрації**

Критерій	Prefect	Mage	Kestra	Dagster
Модель виконання	Гібридна (локальна/хмарна)	Контейнеризована	Універсальна	Універсальна
Підхід до опису робочих процесів	Декоратори Python	Python	YAML	Python
Мови програмування	Python	Python	Python, R, Julia, Node.js, etc.	Python
Спільнота	Велика	Зростаюча	Зростаюча	Зростаюча
Гнучкість	Висока	Середня	Висока	Висока
Масштабованість	Висока	Висока	Висока	Висока
Складність використання	Середня	Низька	Низька	Середня
Підтримка SaaS та хмарних сервісів	Висока	Висока	Середня	Низька
Відображення залежностей	Динамічне	Статичне	Статичне	Статичне
Підтримка контейнеризації	Висока	Висока	Середня	Низька

Висновки

В епоху даних ефективна оркестрація є невід'ємним елементом успішної роботи з даними. Вибір правильного інструменту оркестрації є критичним завданням для будь-якої команди інженерів даних. Кожен інструмент має свої сильні сторони та призначений для різних потреб.

Оптимальний вибір залежить від конкретних вимог, інфраструктури та знайомства команди з інструментом.

Prefect добре підходить , коли в команди не було попереднього досвіду в побудові DAG-ів і потрібний простий в освоєнні інструмент з великою спільнотою, а також потрібна гібридна модель.

Mage підходить для тих випадків, коли потрібна обробка даних в реальному часі та розгортання в контейнеризованих середовищах.

Kestra чудовий вибір для команд, які не мали попереднього досвіду в побудові DAG-ів але мають гарний досвід з розгортанням інфраструктури з застосування принципів IaC.

Dagster добре підходить для команд,. які шукають інструмент для оркестрації dbt процесів та коли надається пріорітет наглядності та простоті у розумінні робочого процесу.

СПИСОК ЛІТЕРАТУРИ

1. Fundamentals of Data Engineering. Authors: Joseph Reis and Matthew Housley -2022. – 447 p.
2. Data Pipelines Pocket Reference: Moving and Processing Data for Analytics. Authors: James Densmore – 2021. – 274 p.
3. Коваленко А. А., Кучук Г. А. Методи синтезу інформаційної та технічної структур системи управління об'єктом критичного застосування. *Сучасні інформаційні системи*. 2018. Т. 2, № 1. С. 22–27. DOI: https://doi.org/10.20998/2522-9052.2018.1.04
4. Свиридов А. С., Коваленко А. А., Кучук Г. А. Метод перерозподілу пропускної здатності критичної ділянки мережі на основі удосконалення ON/OFF-моделі трафіку. *Сучасні інформаційні системи*. 2018. Т. 2, № 2. С. 139–144. DOI: https://doi.org/10.20998/2522-9052.2018.2.24
5. Datsenko, S. and Kuchuk, H. (2023), "Biometric authentication utilizing convolutional neural networks", *Advanced Information Systems*, Vol. 7, no. 2, pp. 87–91, doi: https://doi.org/10.20998/2522-9052.2023.2.12
6. Petrovska, I. and Kuchuk, H. (2023), "Adaptive resource allocation method for data processing and security in cloud environment", *Advanced Information Systems*, Vol. 7, No. 3, pp. 67–73, doi: https://doi.org/10.20998/2522-9052.2023.3.10
7. Офіційний сайт Airflow [Electronic resource] – URL: https://airflow.apache.org/docs/apache-airflow/stable/index.html
8. Офіційний сайт Prefect [Electronic resource] – URL: https://docs.prefect.io/latest/
9. Офіційний сайт Mage [Electronic resource] – URL: https://docs.mage.ai/introduction/overview
10. Офіційний сайт Kestra [Electronic resource] – URL: https://kestra.io/docs
11. Офіційний сайт Dagster [Electronic resource] – URL: https://docs.dagster.io/getting-started/what-why-dagster

Received (Надійшла) 22.02.2024
Accepted for publication (Прийнята до друку) 17.04.2024

Modern data orchestration tools for building big data processing pipelines

V. Zaleskyi, P. Ivanovskii, V. Fedorchenko

Abstract. The data universe in modern companies is constantly expanding. As the amount of data grows, so does the need for management, schedule synchronization, and processing challenges. Companies need to break down the barriers between data sources and storage in order to truly utilize all of the information they collect. Data orchestration allows organizations to automate and optimize their data, transforming it into operational assets so that valuable insights can be used for real-time business decision making. By some estimates, 80% of the work involved in data analysis is spent on data collection and preparation, meaning that data orchestration can significantly reduce the amount of time spent on processing and scheduling. **The goal of this work** to overview modern data orchestration tools. **The object of research** is data engineering. **The subject of research** is data orchestration. **Conclusions.** In the age of big data, effective orchestration is an integral part of successful data work. Choosing the right orchestration tool is a critical task for any data engineering team. Each tool has its own strengths and is designed for different needs. The optimal choice depends on the specific requirements, infrastructure, and the team's familiarity with the tool.

Keywords: data orchestration, data processing pipeline, ETL, DAG.